

# Sonic Visualiser




Craig Stuart Sapp

14 April 2016

# Website

<http://www.sonicvisualiser.org>

<http://www.sonicvisualiser.org/download.html>

	Linux	OS/X	Windows
Version 2.5	 <b>Download</b> 64-bit Linux package for Ubuntu 14.04	 <b>Download</b> 64-bit Intel binary for OS/X 10.6 or newer	 <b>Download</b> Windows installer
	For older versions, other platforms, and source code, see the <a href="#">Sonic Visualiser code project download page</a> .		



**SONIC VISUALISER**

Sonic Visualiser is an application for viewing and analysing the contents of music audio files.

22nd October 2015: Sonic Visualiser 2.5 has been released. Get it [here!](#)

The aim of Sonic Visualiser is to be the first program you reach for when want to study a musical recording rather than simply listen to it.

We hope Sonic Visualiser will be of particular interest to musicologists, archivists, signal-processing researchers and anyone else looking for a friendly way to take a look at what lies inside the audio file.

Sonic Visualiser is Free Software, distributed under the [GNU General Public License](#) (v2 or later) and available for Linux, OS/X, and Windows. It was developed at the [Centre for Digital Music](#) at Queen Mary, University of London.

*Citations:* If you are using Sonic Visualiser in research work for publication, please cite ([pdf](#) | [bib](#)) Chris Cannam, Christian Landone, and Mark Sandler, *Sonic Visualiser: An Open Source Application for Viewing, Analysing, and Annotating Music Audio Files*, in Proceedings of the ACM Multimedia 2010 International Conference.

Developed at the Centre for Digital Music, Queen Mary, University of London.

Partially funded by the EPSRC through the OMRAS2 project EP/E017614/1.

Partially funded by the European Commission through the SIMAC project IST-FP6-507142.

Partially funded by the Arts and Humanities Research Council through the Research Centre for the History and Analysis of Recorded Music.

**OMRAS II**

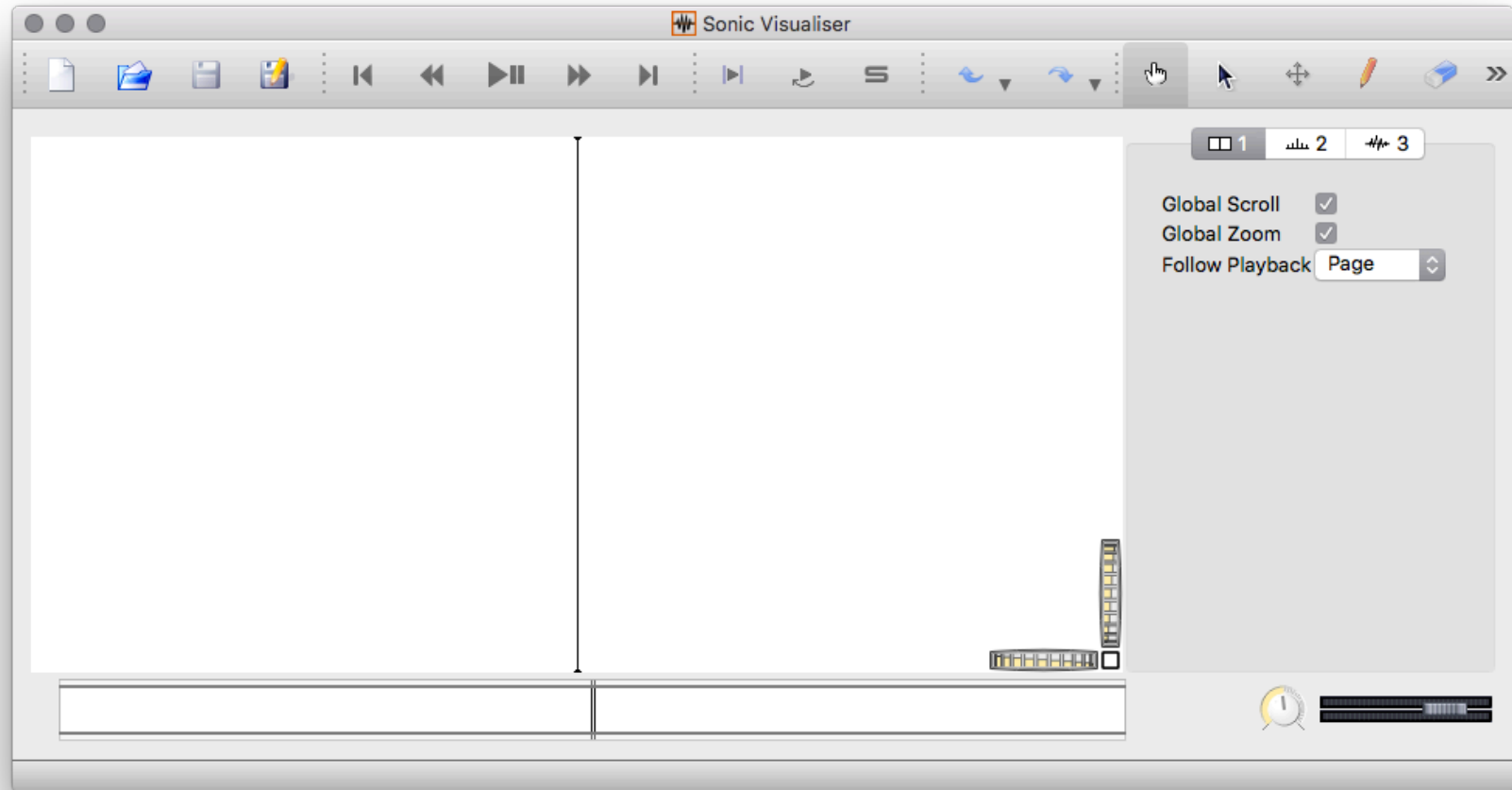
**simac**  
www.semanticsaudio.org

**EASAIER**  
Enabling Access to Shared Archives

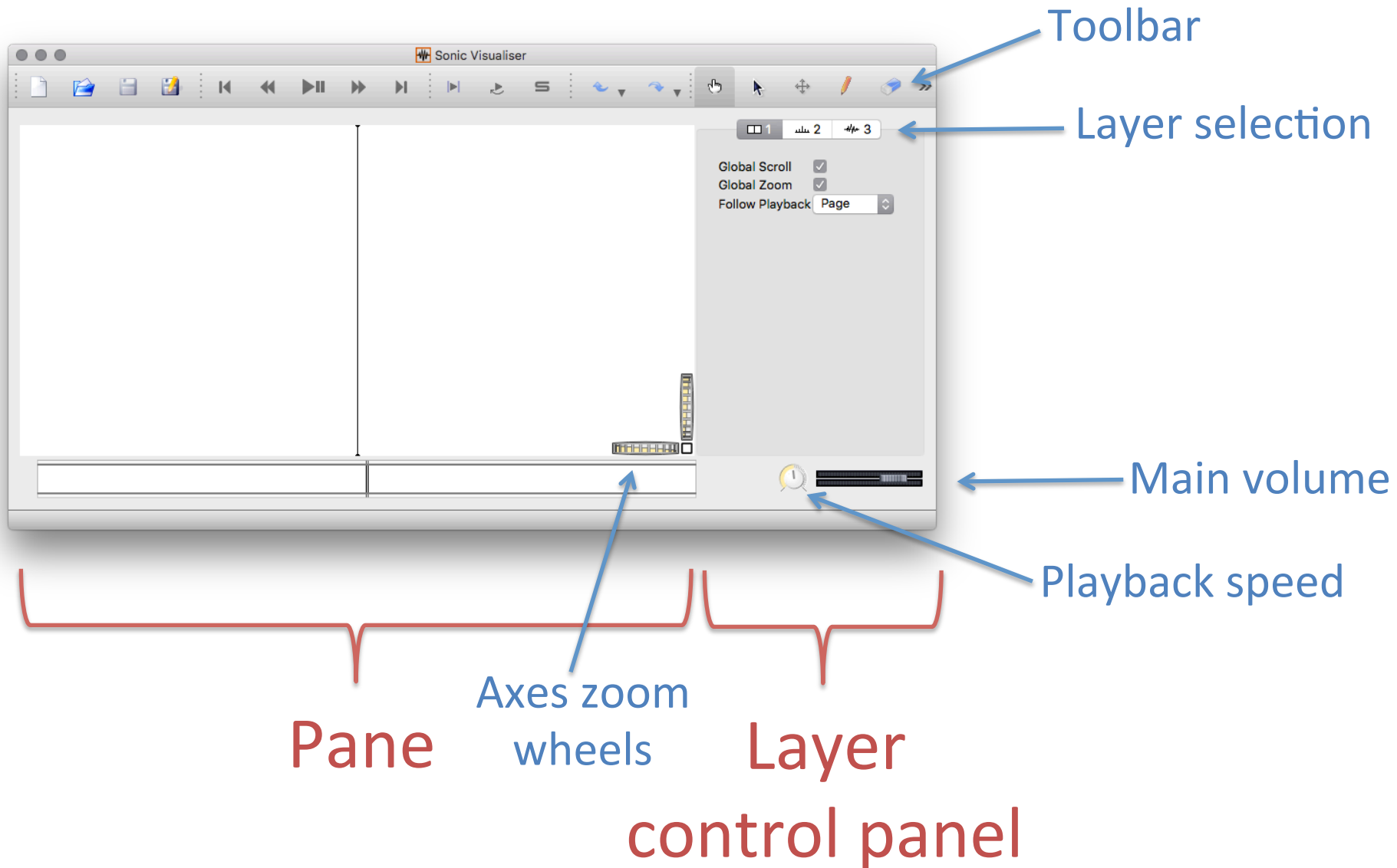
**CHARM**  
Research Centre for the History and Analysis of Recorded Music

HOME  
FEATURES  
SCREENSHOTS  
DOWNLOAD  
COMMUNITY  
DOCUMENTATION  
VIDEOS  
VAMP PLUGINS

# SV window

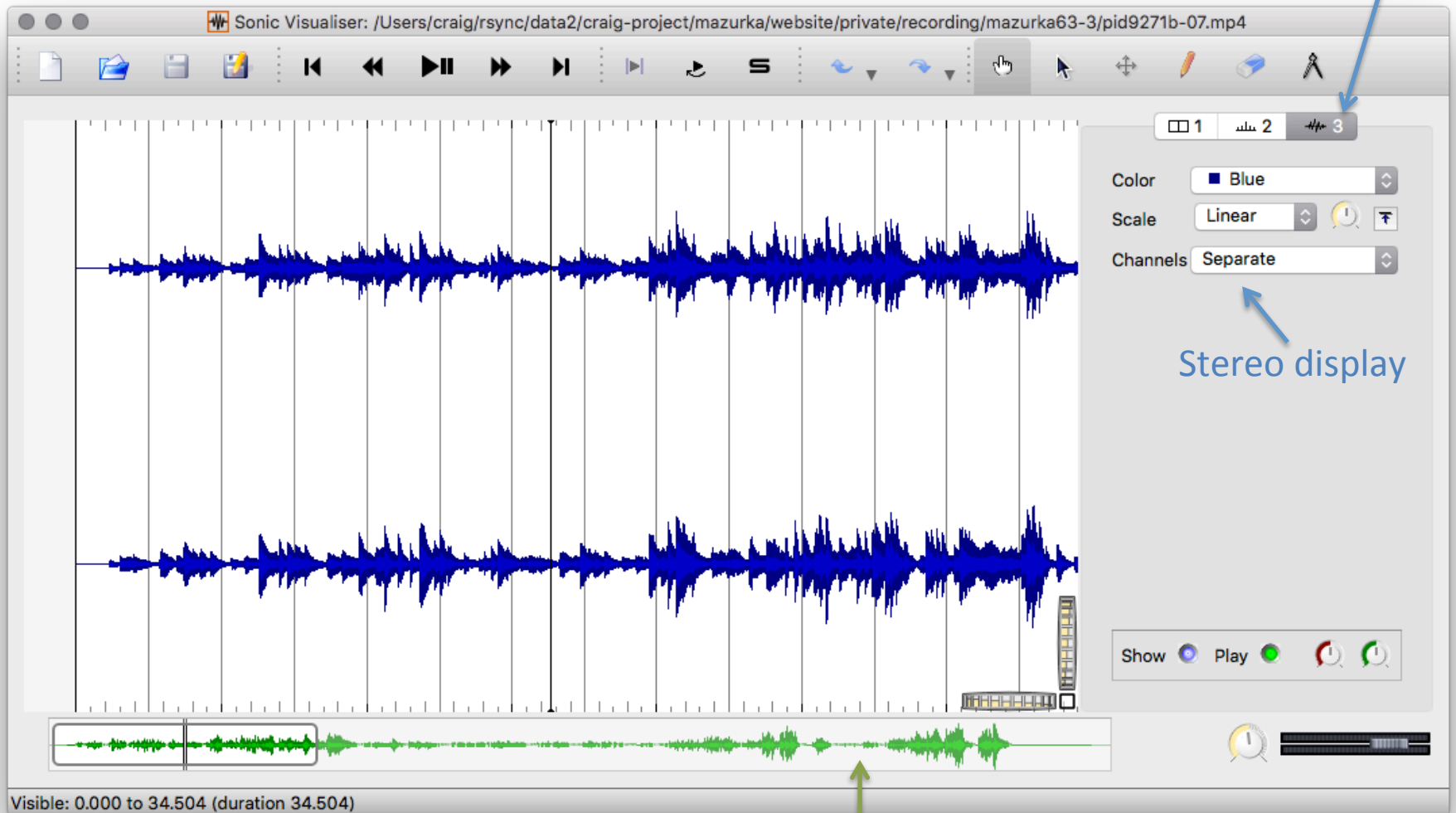


# SV window main components



# Waveform layer

Waveform layer controls



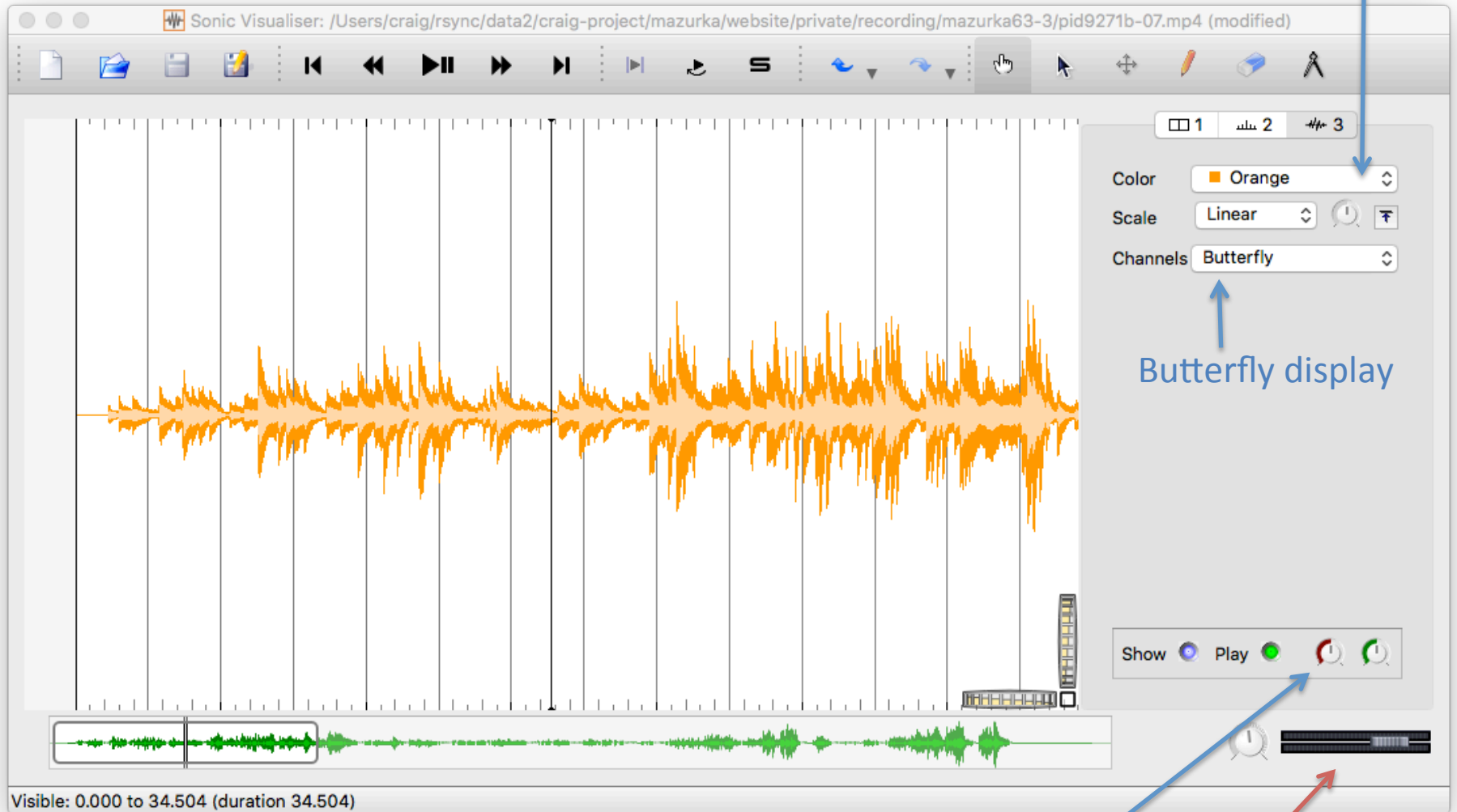
Stereo display

Currently visible audio

Waveform navigator

# Waveform styling

Waveform color



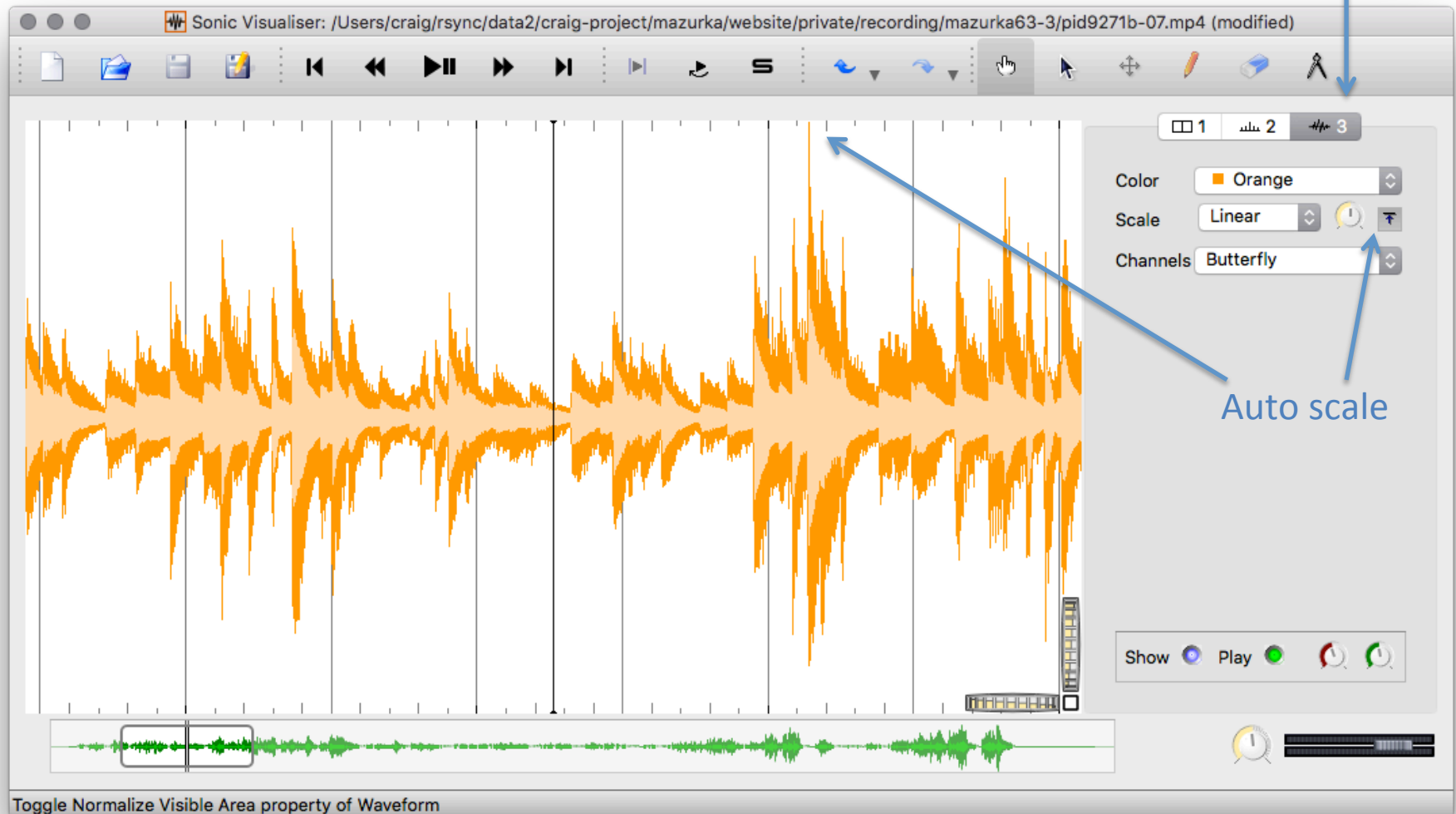
Butterfly display

Layer volume

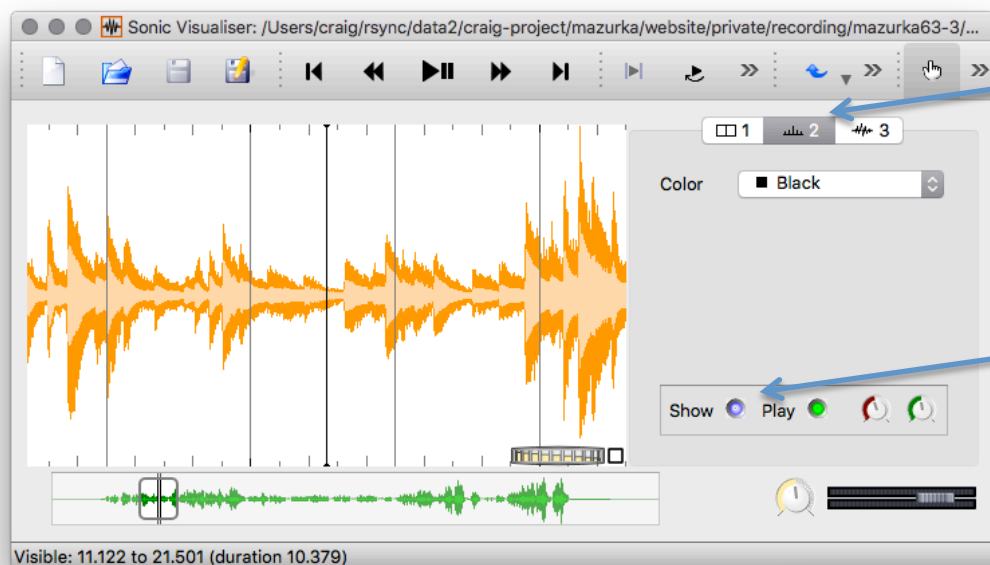
Master volume

# Waveform styling

Active layer  
Is highlighted

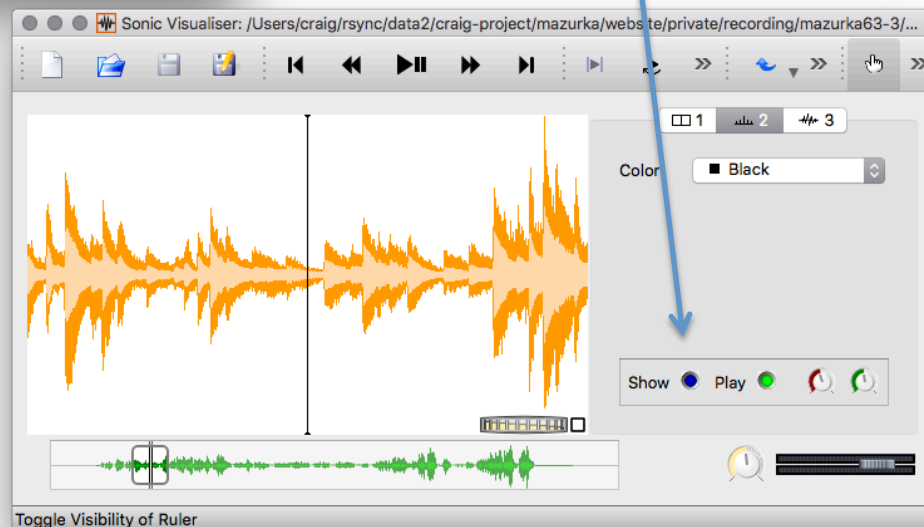


# Time-ruler layer



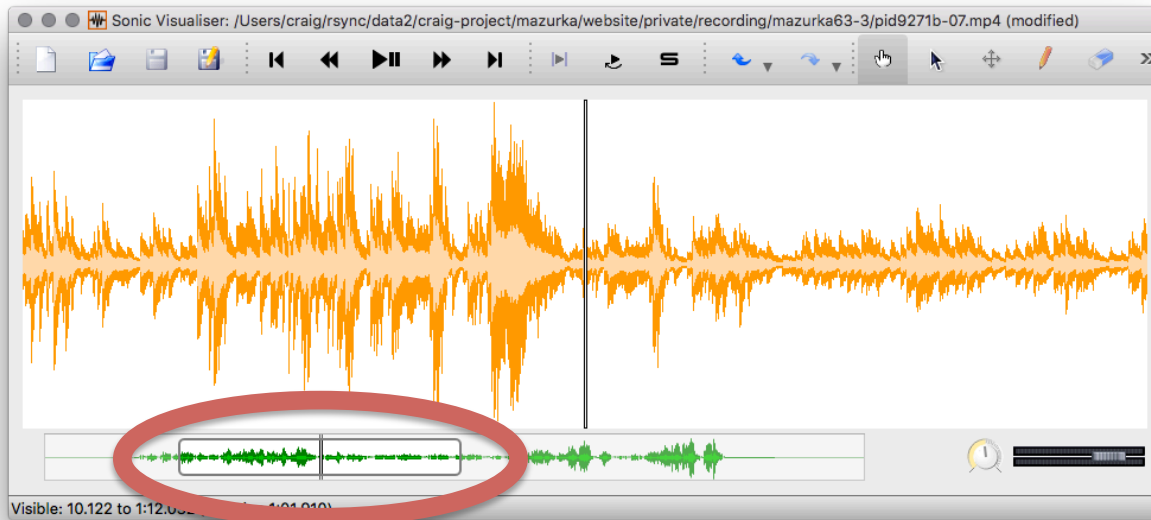
Time layer  
is selected

Show/hide  
layer

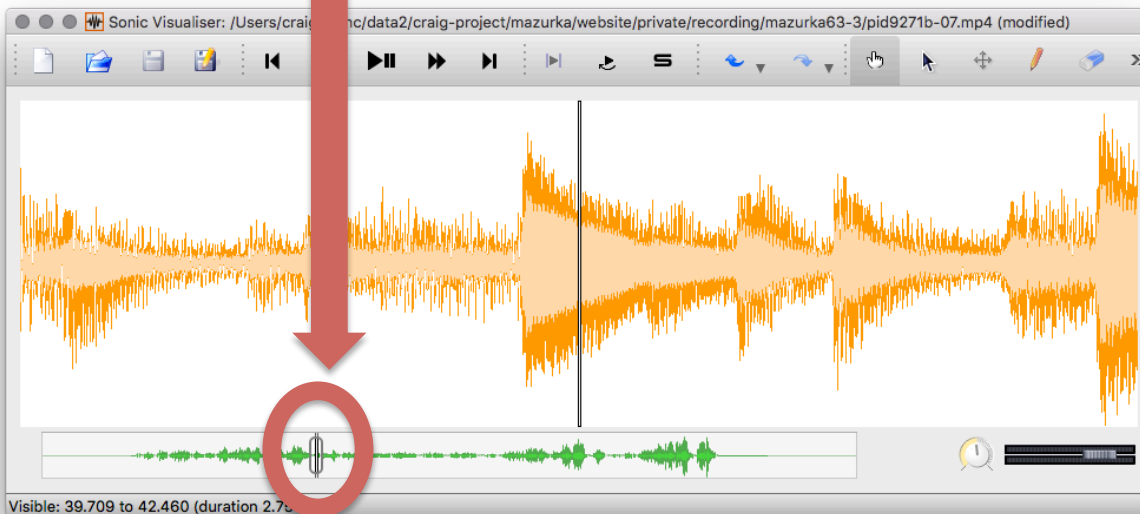




# Zoom in/out with arrow keys

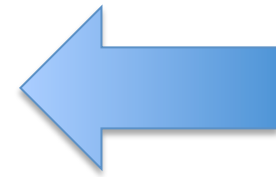
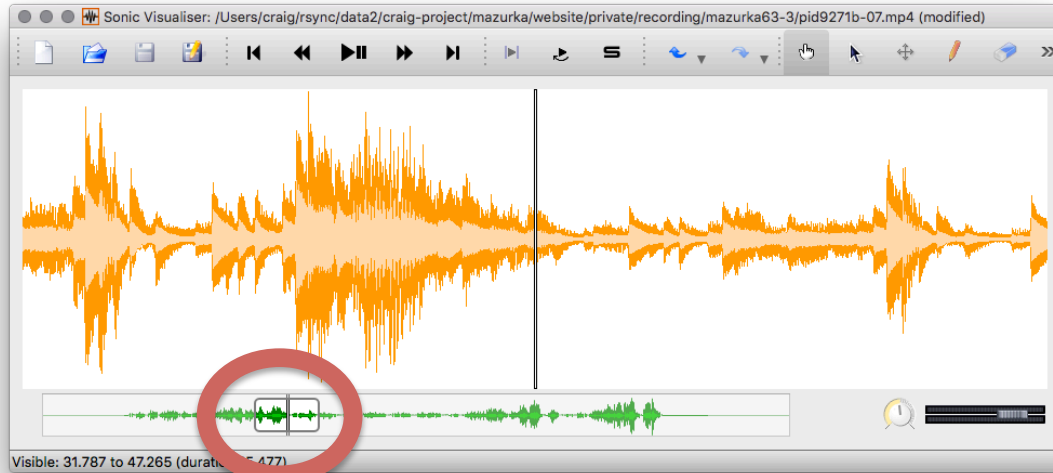


Zoom  
out

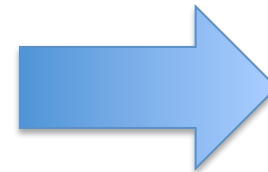
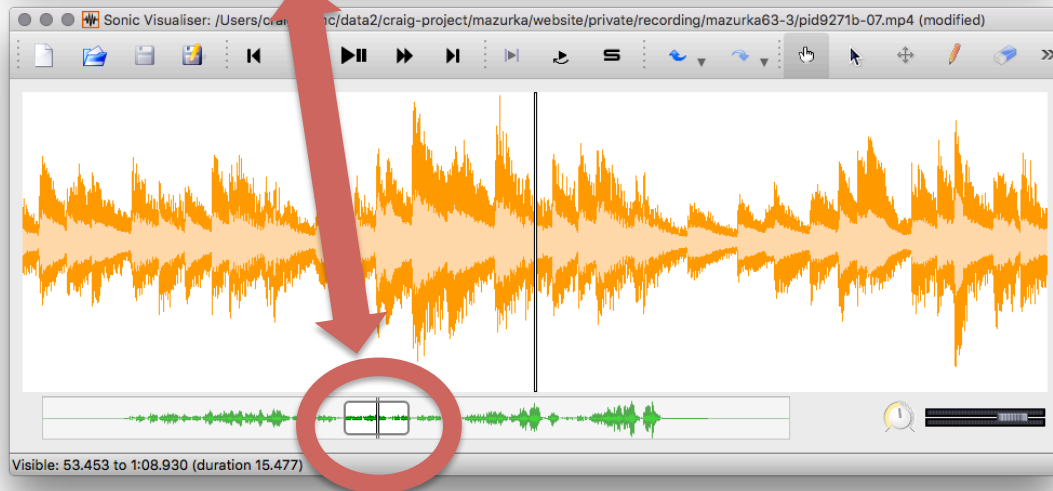


Zoom  
in

# Panning with arrow keys

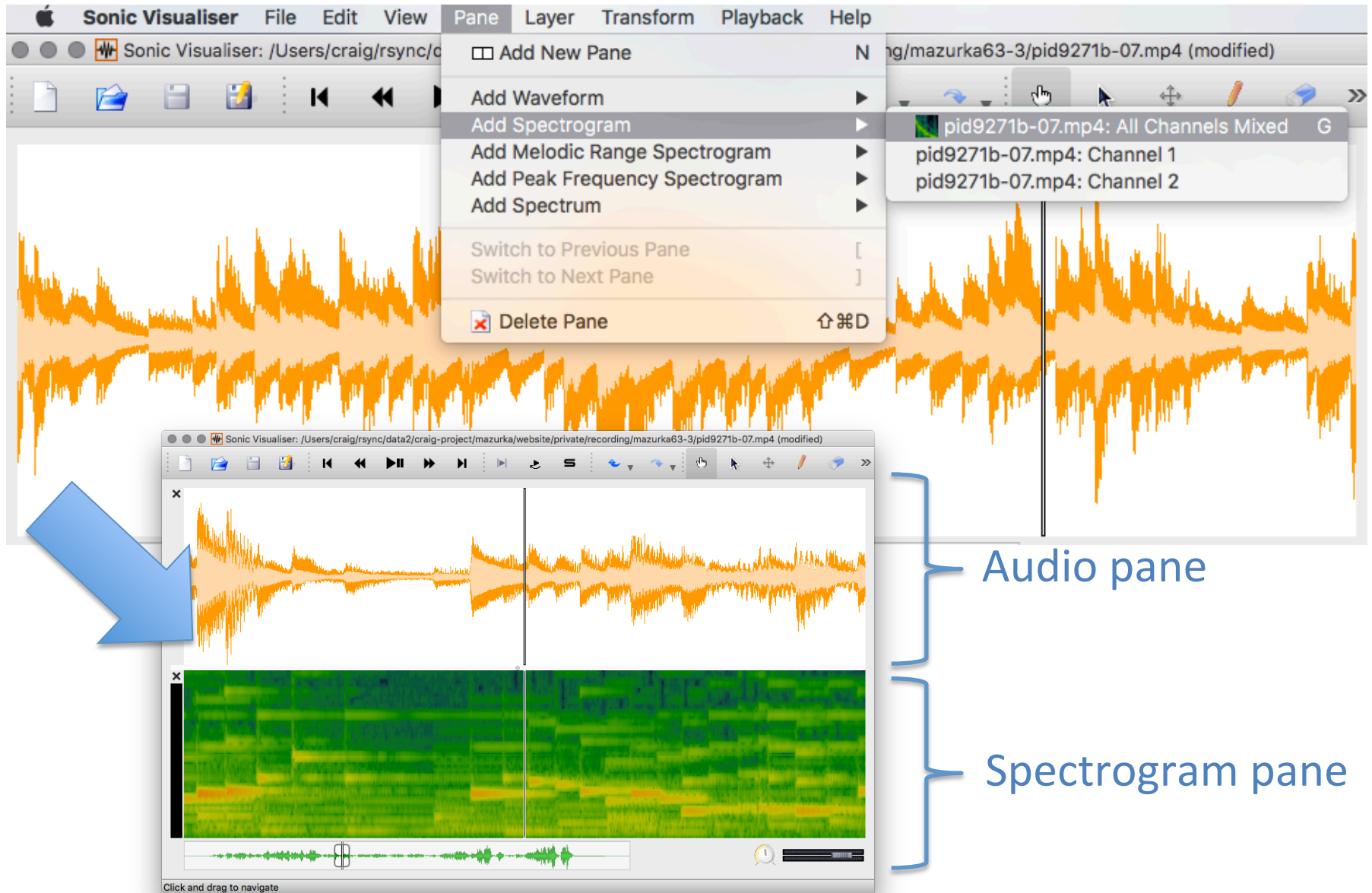


Move  
left

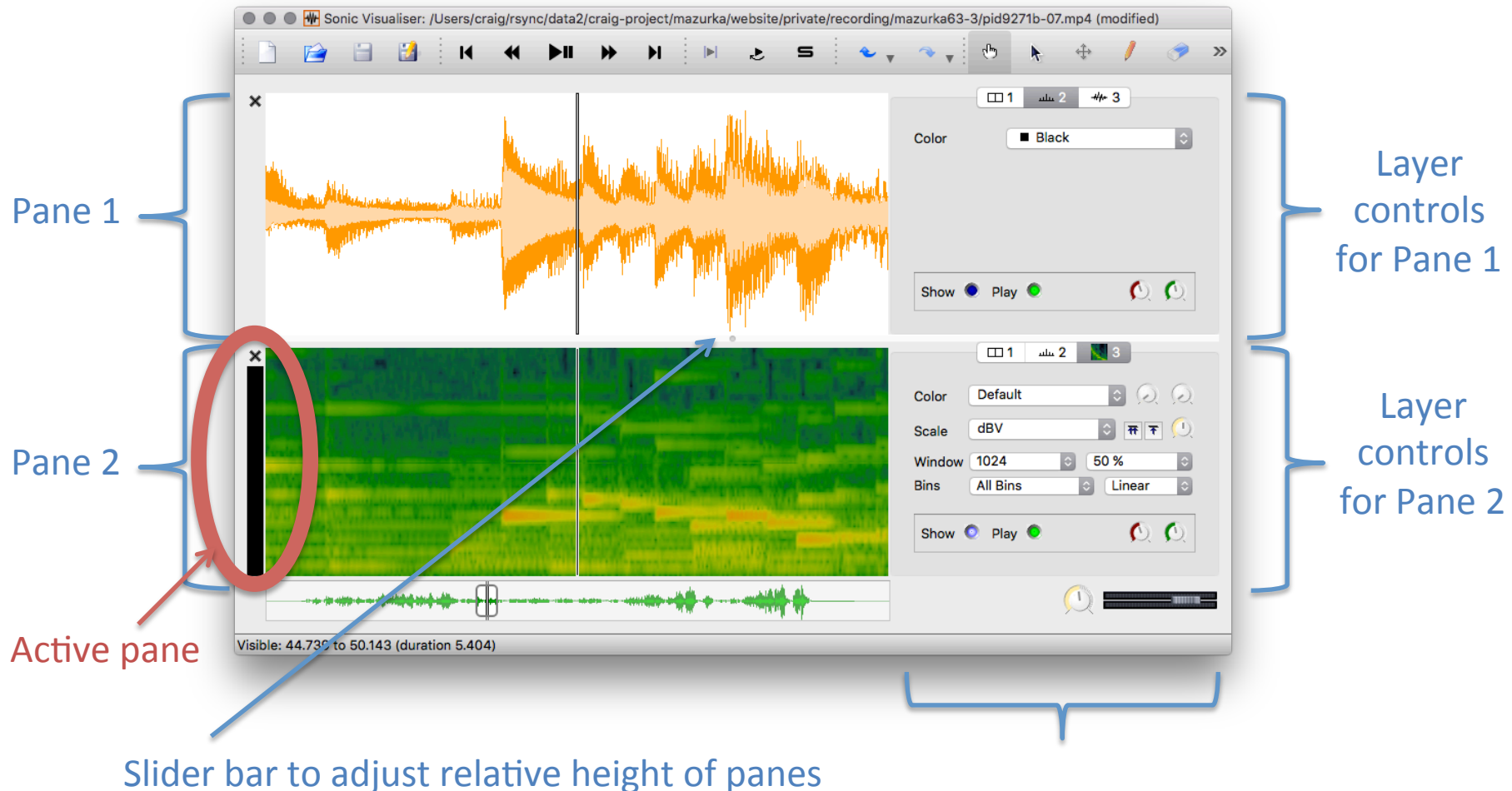


Move  
right

# Spectrogram



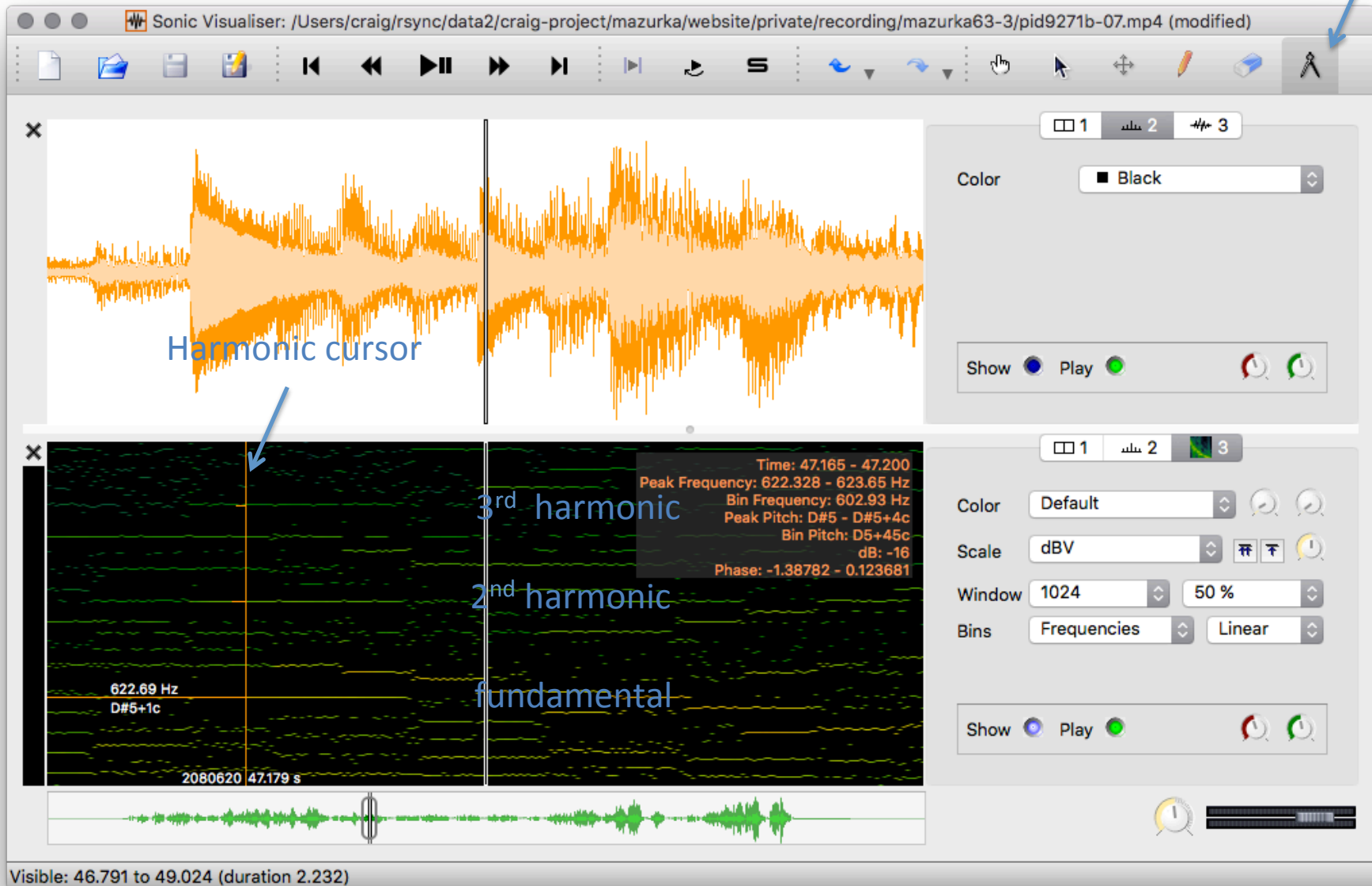
# Panes vs. Layers



X key toggles control panel displays

# Measuring pitch of a note

Show harmonic  
cursor



# Reading pitch



Time: 47.211 - 47.246  
Peak Frequency: 622.84 - 623.381 Hz  
Bin Frequency: 602.93 Hz  
Peak Pitch: D#5+2c - D#5+3c  
Bin Pitch: D5+45c  
dB: -16  
Phase: -1.84294 - -0.351101

Ignore “Bin Frequency” and “Bin Pitch”. These are not physical measurements, but rather related to analysis window settings.

Musical Pitch is “Peak Pitch” field

D#5+2c – D#5+3c

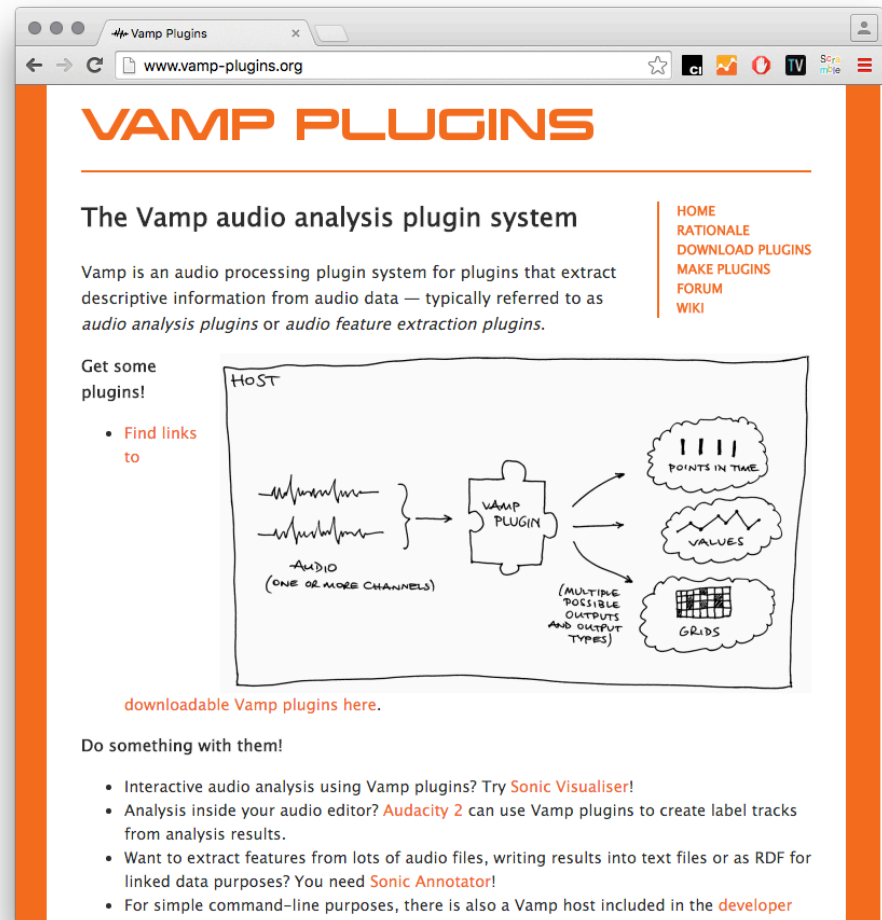
Meaning: D#5 (15 semitones above middle C), in the range from +2 center to +3 cents from Equal tempered tuning referenced at A440.

Corresponds to “Peak Frequency: 622.84-623.381 Hz”

# Vamp Plugins

<http://www.vamp-plugins.org>

Audio processing tools  
accessed from the “Transform”  
menu at the top of the Sonic  
Visualiser Window.



The screenshot shows the Vamp Plugins website in a web browser. The page has an orange header with the title "VAMP PLUGINS". Below the header, there's a section titled "The Vamp audio analysis plugin system" with a description of Vamp as an audio processing plugin system. To the right of this section is a sidebar with links: HOME, RATIONALE, DOWNLOAD PLUGINS, MAKE PLUGINS, FORUM, and WIKI. Below the description, there's a section "Get some plugins!" with a link "Find links to". A diagram titled "Host" shows audio input (one or more channels) entering a "VAMP PLUGIN" box, which then outputs to three cloud shapes labeled "POINTS IN TIME", "VALUES", and "GRIDS". Below the diagram is a link "downloadable Vamp plugins here.". At the bottom, there's a section "Do something with them!" with a list of bullet points: "Interactive audio analysis using Vamp plugins? Try Sonic Visualiser!", "Analysis inside your audio editor? Audacity 2 can use Vamp plugins to create label tracks from analysis results.", "Want to extract features from lots of audio files, writing results into text files or as RDF for linked data purposes? You need Sonic Annotator!", and "For simple command-line purposes, there is also a Vamp host included in the developer".

**VAMP PLUGINS**

## The Vamp audio analysis plugin system

Vamp is an audio processing plugin system for plugins that extract descriptive information from audio data — typically referred to as *audio analysis plugins* or *audio feature extraction plugins*.

Get some plugins!

- Find links to

downloadable Vamp plugins here.

Do something with them!

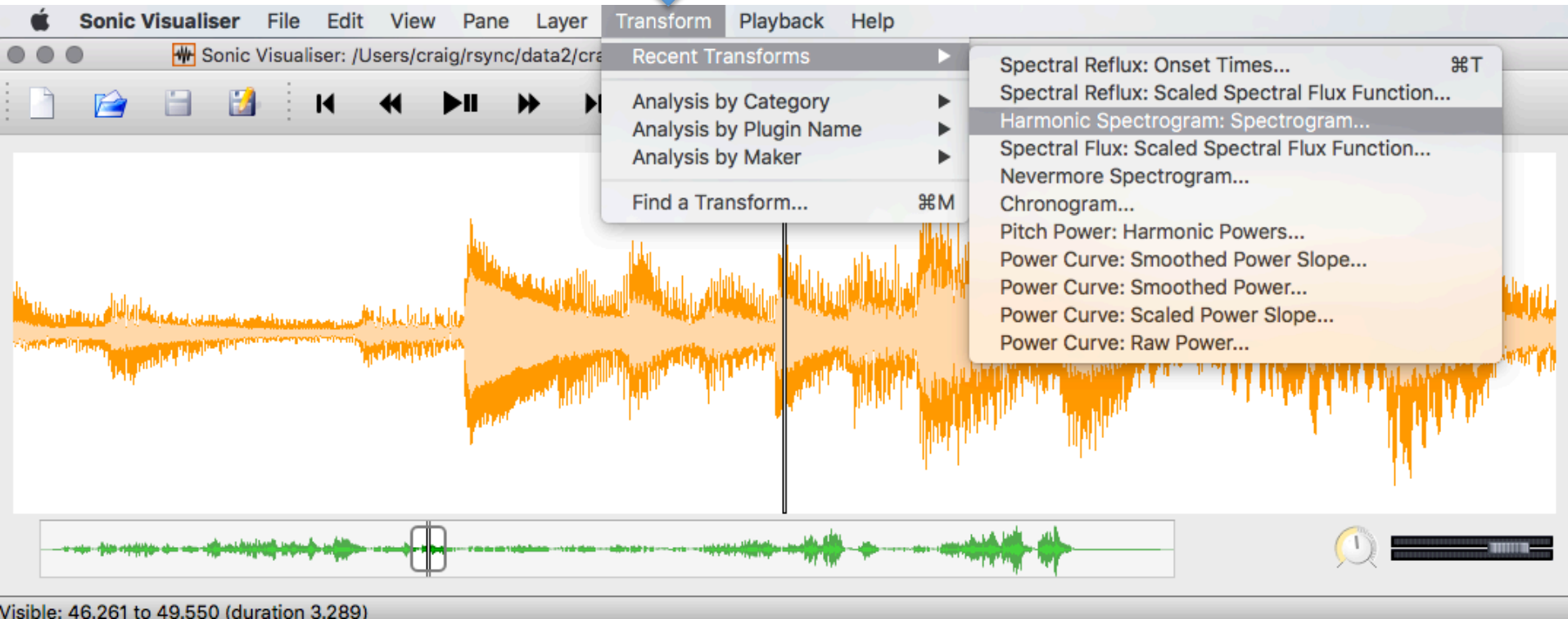
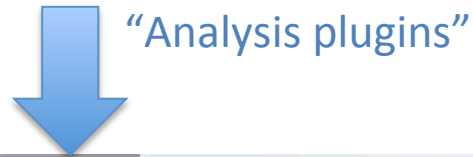
- Interactive audio analysis using Vamp plugins? Try [Sonic Visualiser](#)!
- Analysis inside your audio editor? [Audacity 2](#) can use Vamp plugins to create label tracks from analysis results.
- Want to extract features from lots of audio files, writing results into text files or as RDF for linked data purposes? You need [Sonic Annotator](#)!
- For simple command-line purposes, there is also a Vamp host included in the [developer](#)

**Host**

Audio (one or more channels) → VAMP PLUGIN → (multiple possible outputs and output types)

POINTS IN TIME  
VALUES  
GRIDS

# Transforms menu





# Vamp Plugin installation

<http://www.vamp-plugins.org/download.html#install>

- OS/X plugins end in “.dylib” and are placed in ~/Library/Audio/PlugIns/Vamp folder
- Windows 64-bit plugins end in “dll” and placed in C:\Program Files\Vamp Plugins
- Windows 32-bit plugins end in “.dll” and placed in C:\Program Files (x86)\Vamp Plugins
- Linux plugins end in “.so” and are placed in ~/vamp

32-bit plugins only work in 32-bit Sonic Visualiser

64-bit plugins only work in 64-bit Sonic Visualiser

# Mazurka Plugins

<http://sv.mazurka.org.uk>

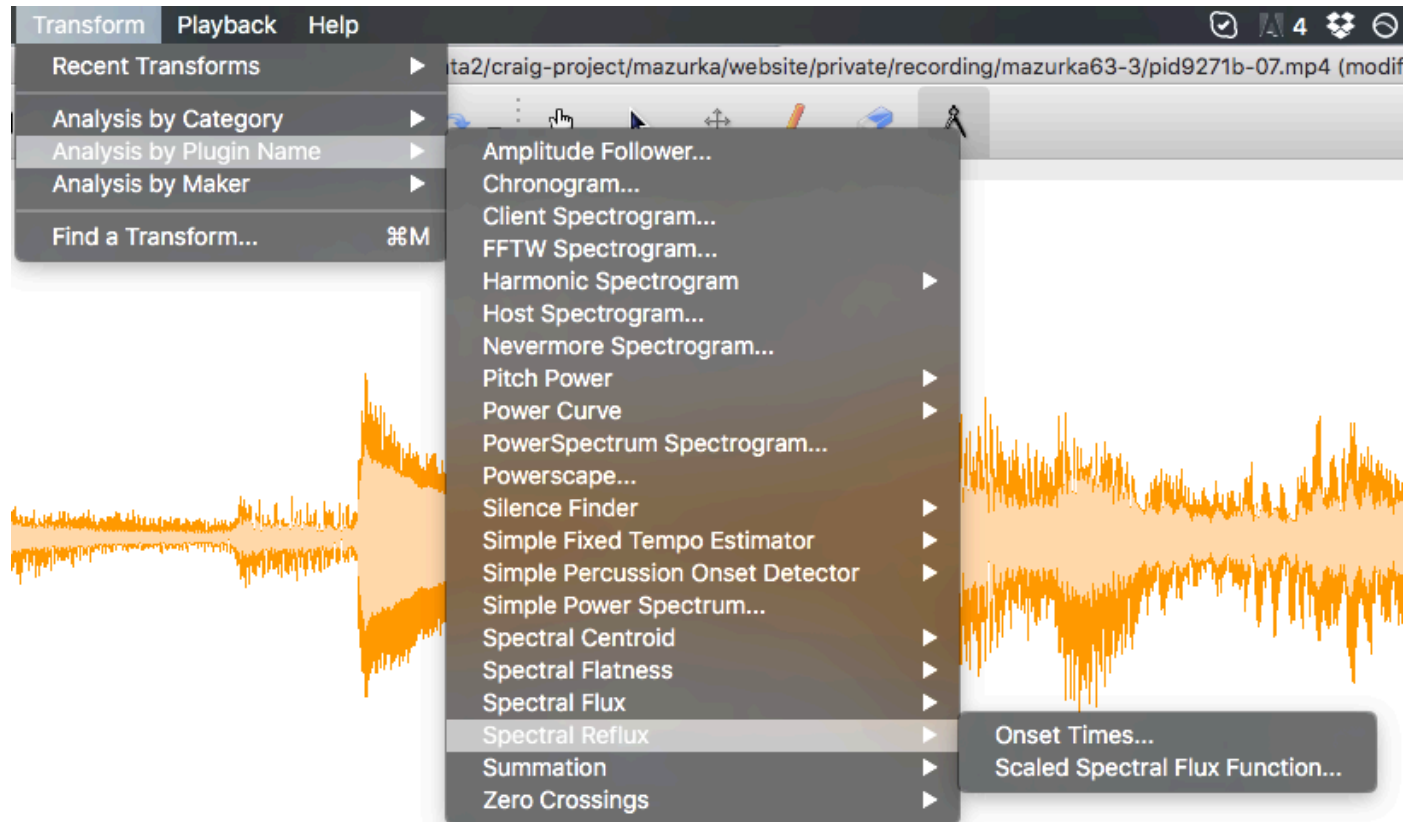


- Designed for performance data extraction in piano music  
(for the Mazurka Project at CHARM (2005–2007))
- Main data-entry plugins:
  - **MzSpectralReflux** – note onset detector
  - **MzHarmonicSpectrum** – spectrum with de-emphasis of harmonics
  - **MzPowerCurve** – loudness measurements (used instead of spectral reflux plugin in noisy recordings)
- Basic discussion of data entry at:  
<http://wiki.ccarh.org/wiki/Op27>

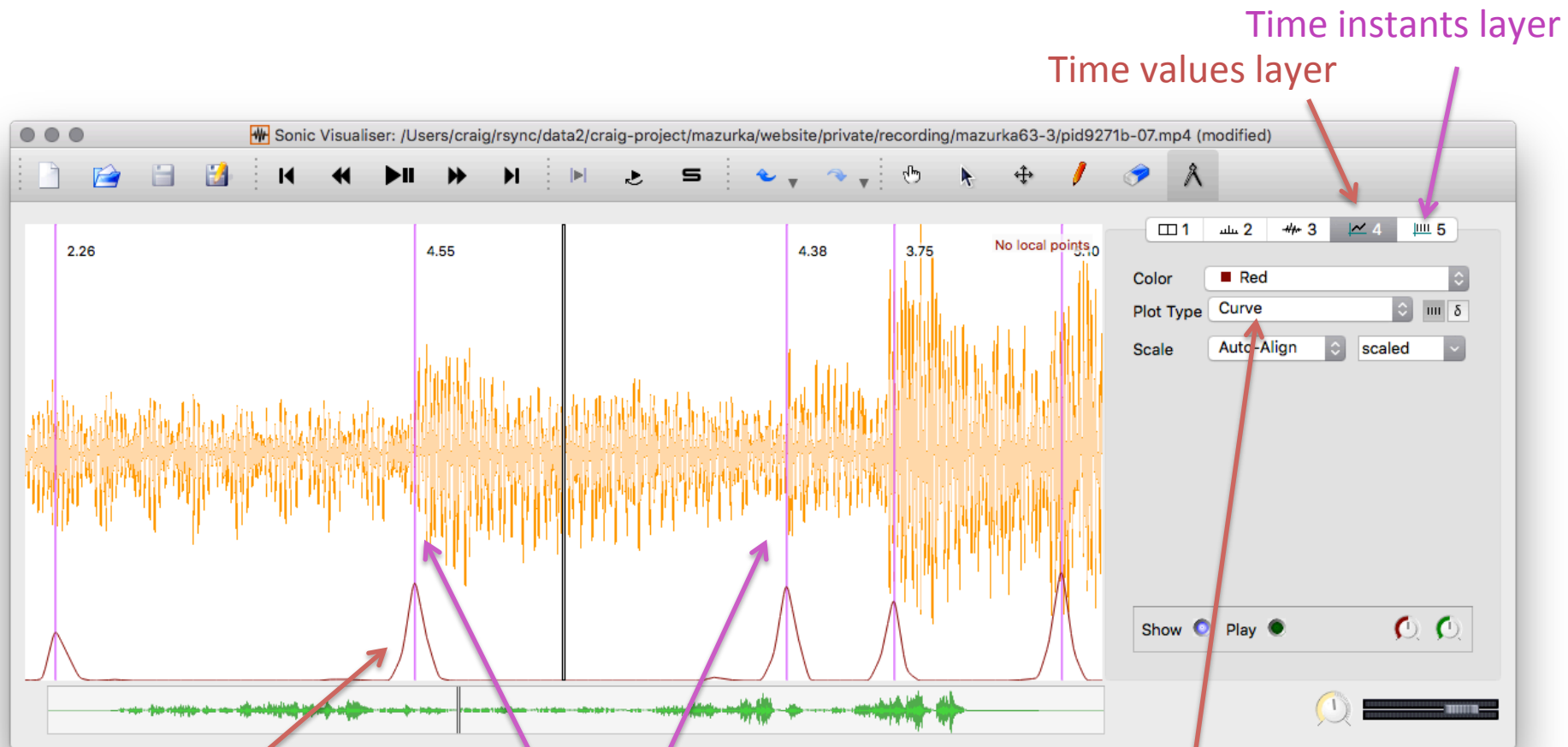
# SpectralReflux plugin

## Two outputs:

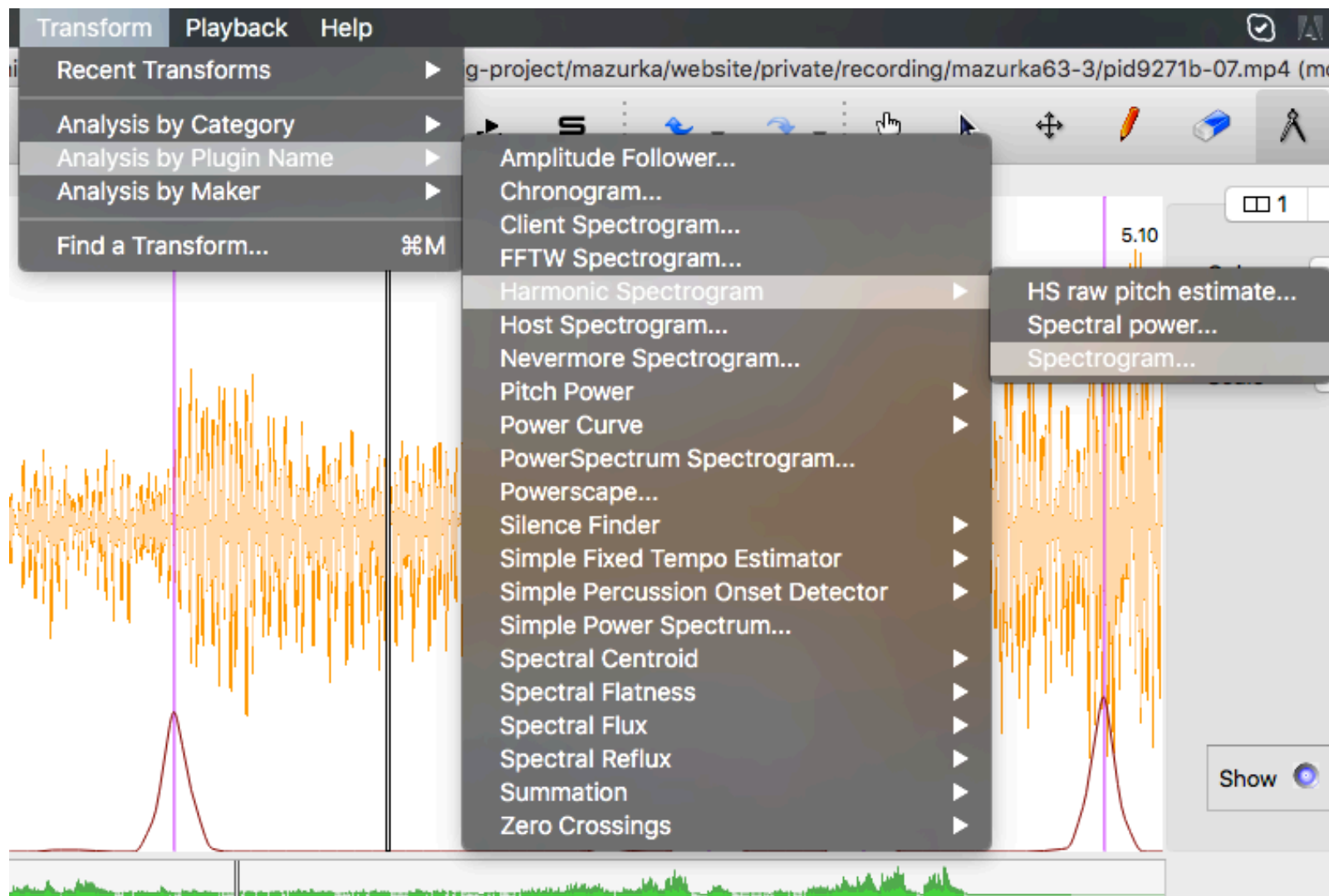
1. “Onset Times” – generates a “time instant” layer with marks at start of notes.
2. “Scaled Spectral Flux Fuction” – generates “Time values” layer which is the flux function with peaks being the detected onsets.



# SpectralReflux plugin outputs



# Harmonic Spectrogram plugin

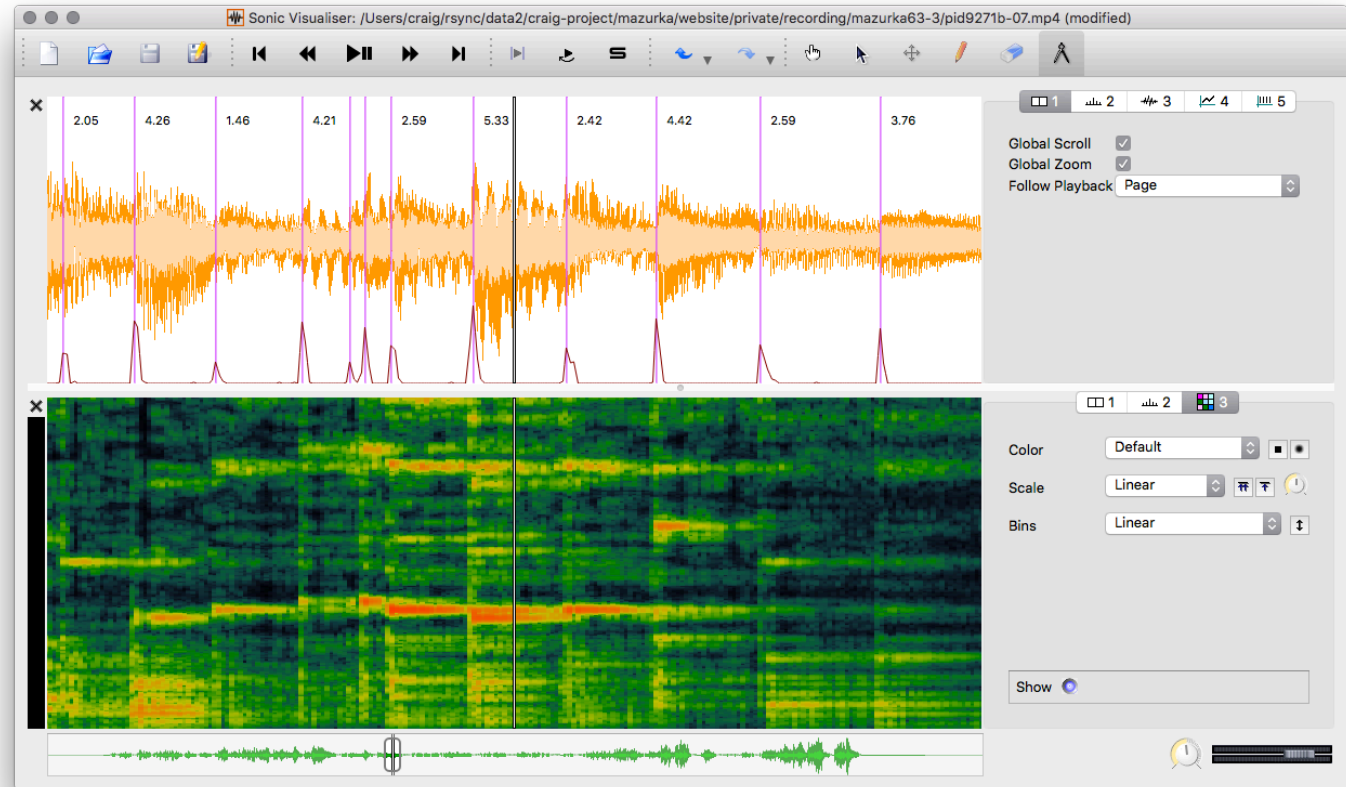


# Harmonic Spectrogram: spectrogram output

1) Create new pane  
(Pane → Add new pane)

2) Run Harmonic  
Spectrogram plugin  
with output  
“spectrogram”

3) Set the scale to  
“Linear” in the  
layer controls  
(usually looks better than “Log” scale).

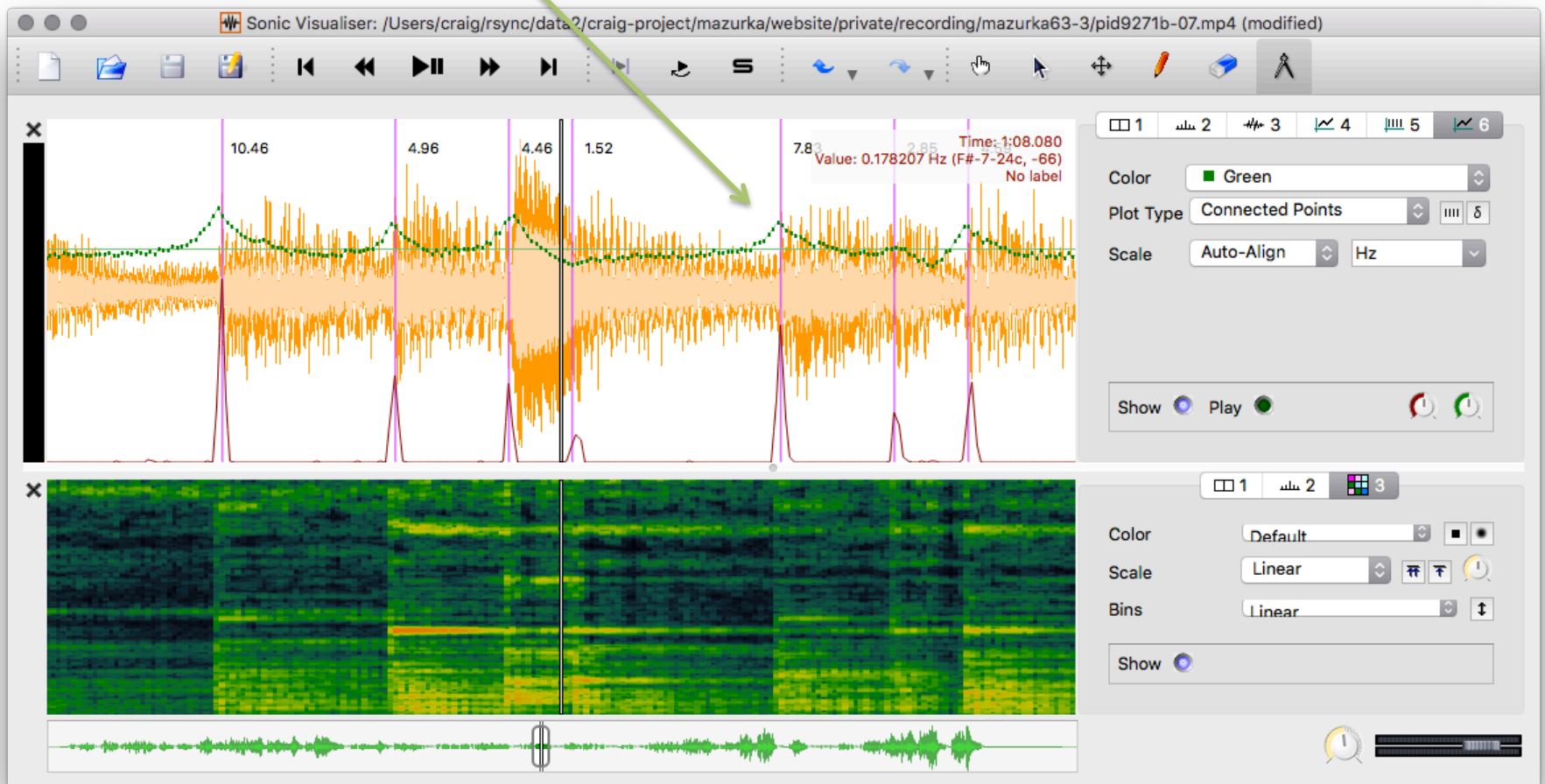


HS is useful for following the melody line and keeping track of basic pitch information.

# Noisy recordings

For noisy recordings, use PowerCurve: scaled power slope plugin is useful for onset IDs.

PowerCurve: scaled power slope



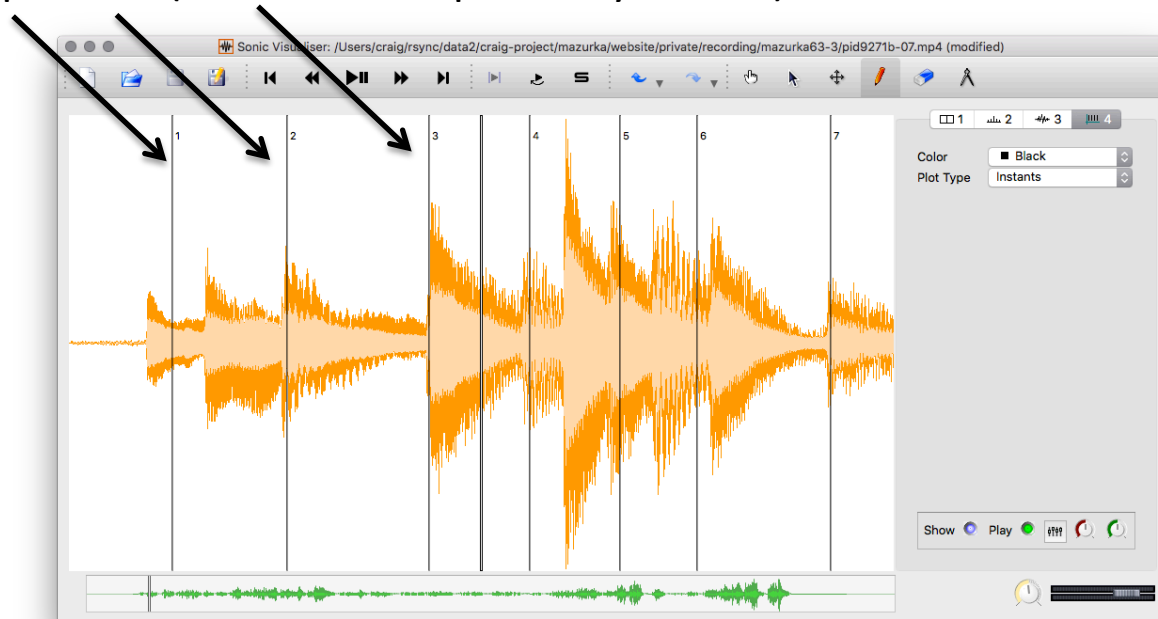
# Tapping

Users can insert time instants while listening to the audio.

Press “;” key to insert a time instant at the current audio position.

Usually best to create empty time instant layer first (but one will be auto-created at time of first tap).

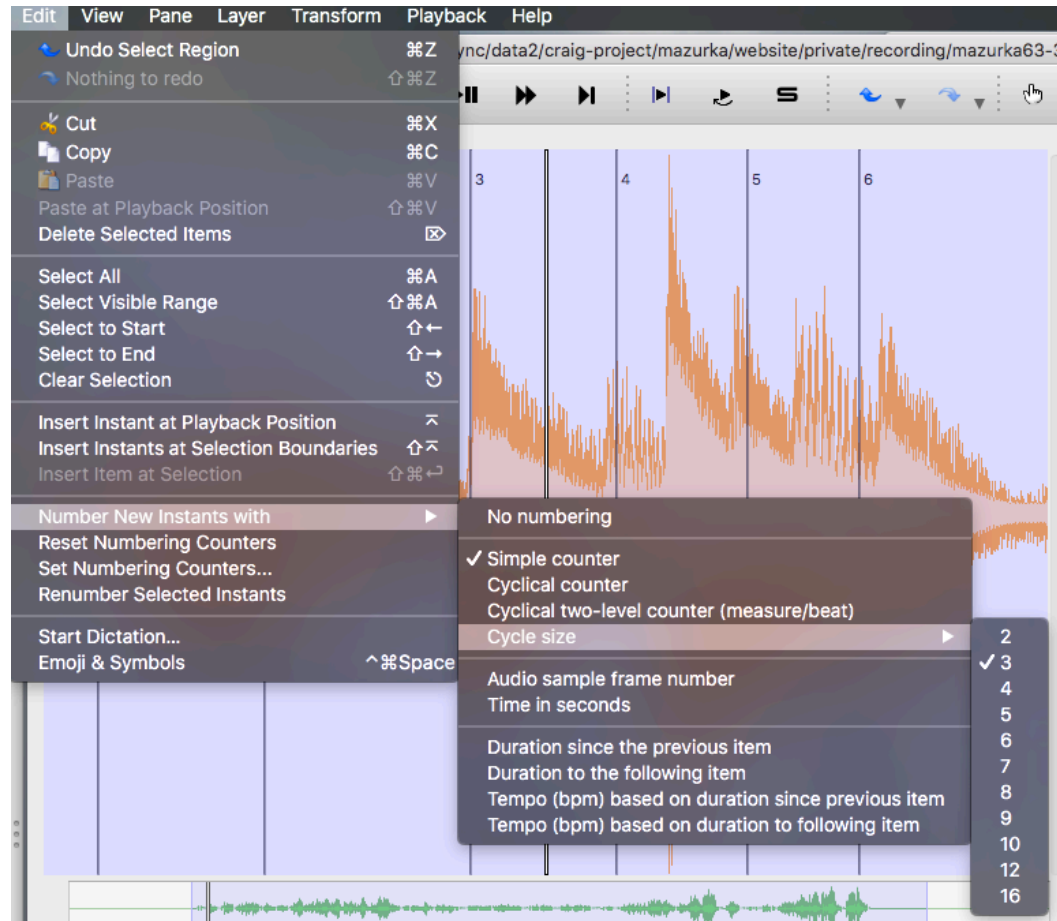
Tap marks (numbered sequentially from 1)





# Cyclical counter labels

Choose cycle size: Edit → Number New Instants with → Cycle size → 3



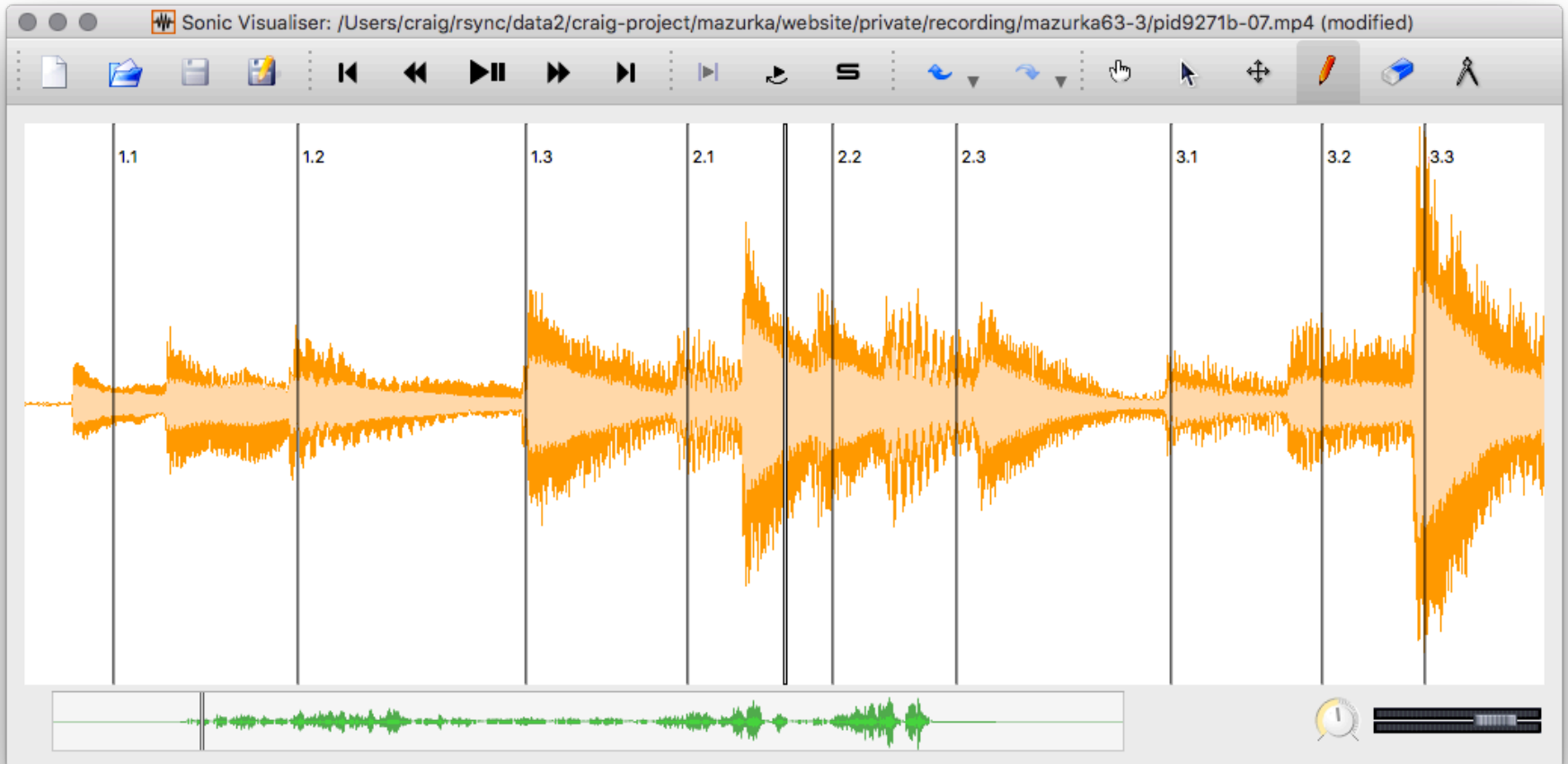
# Cyclical counter labels

Choose cycle size: Edit → Number New Instants with → Cycle size → 3

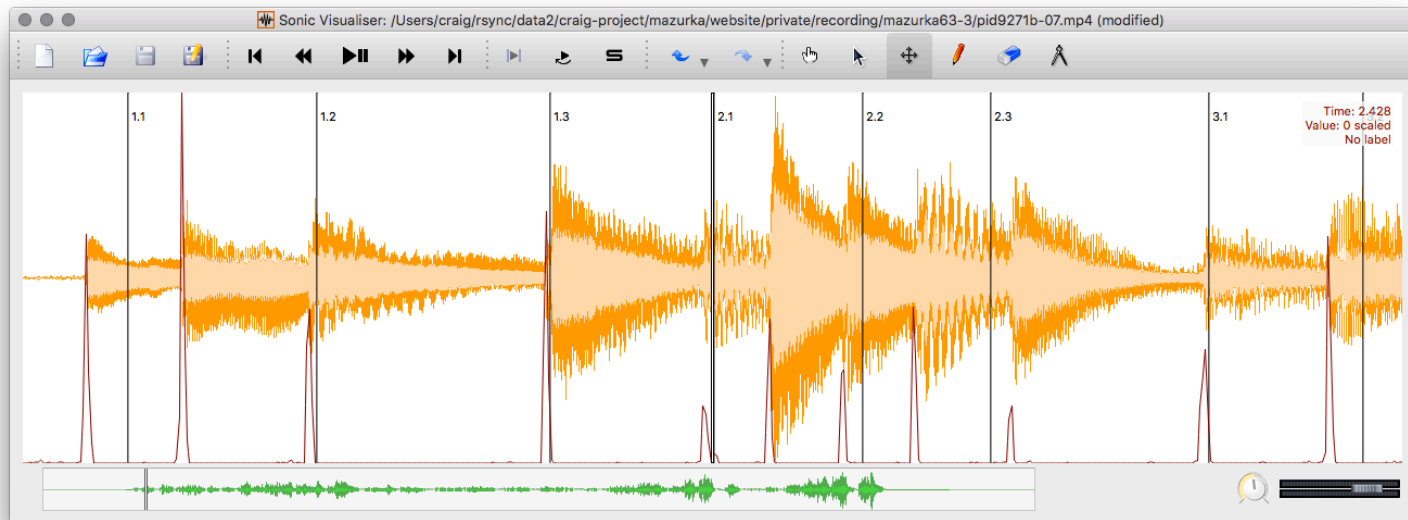
Then select: Edit → Number New Instants → Cyclical two-level counter (measure/beat)

Then select all time instants in layer (Control-A or Command-A)

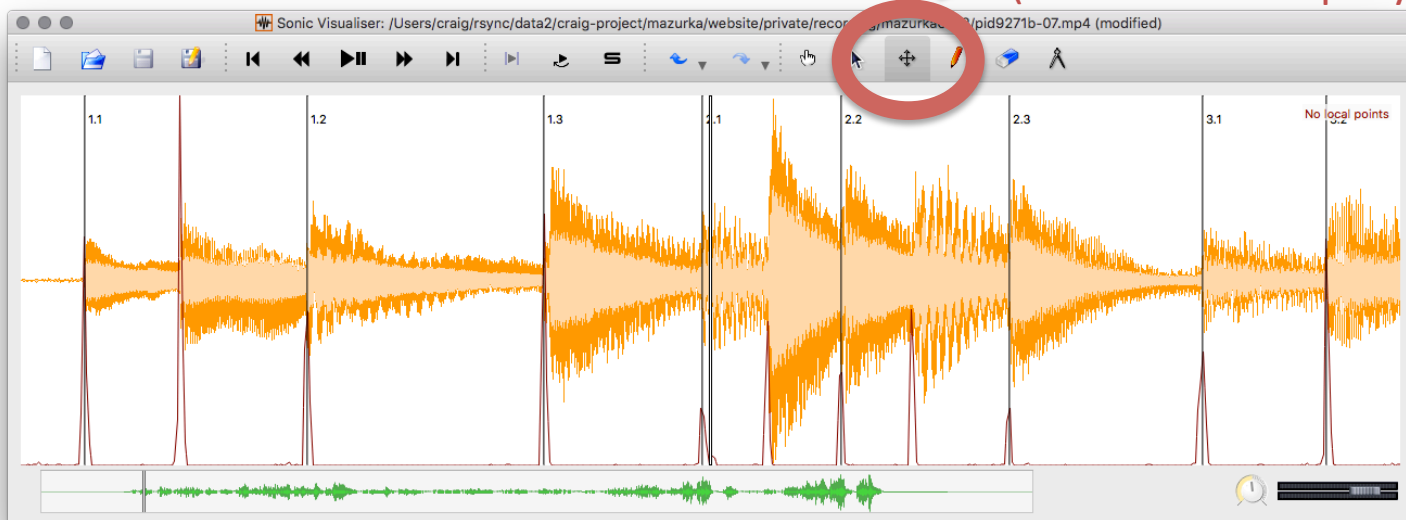
Then choose: Edit → Renumber Selected Instants



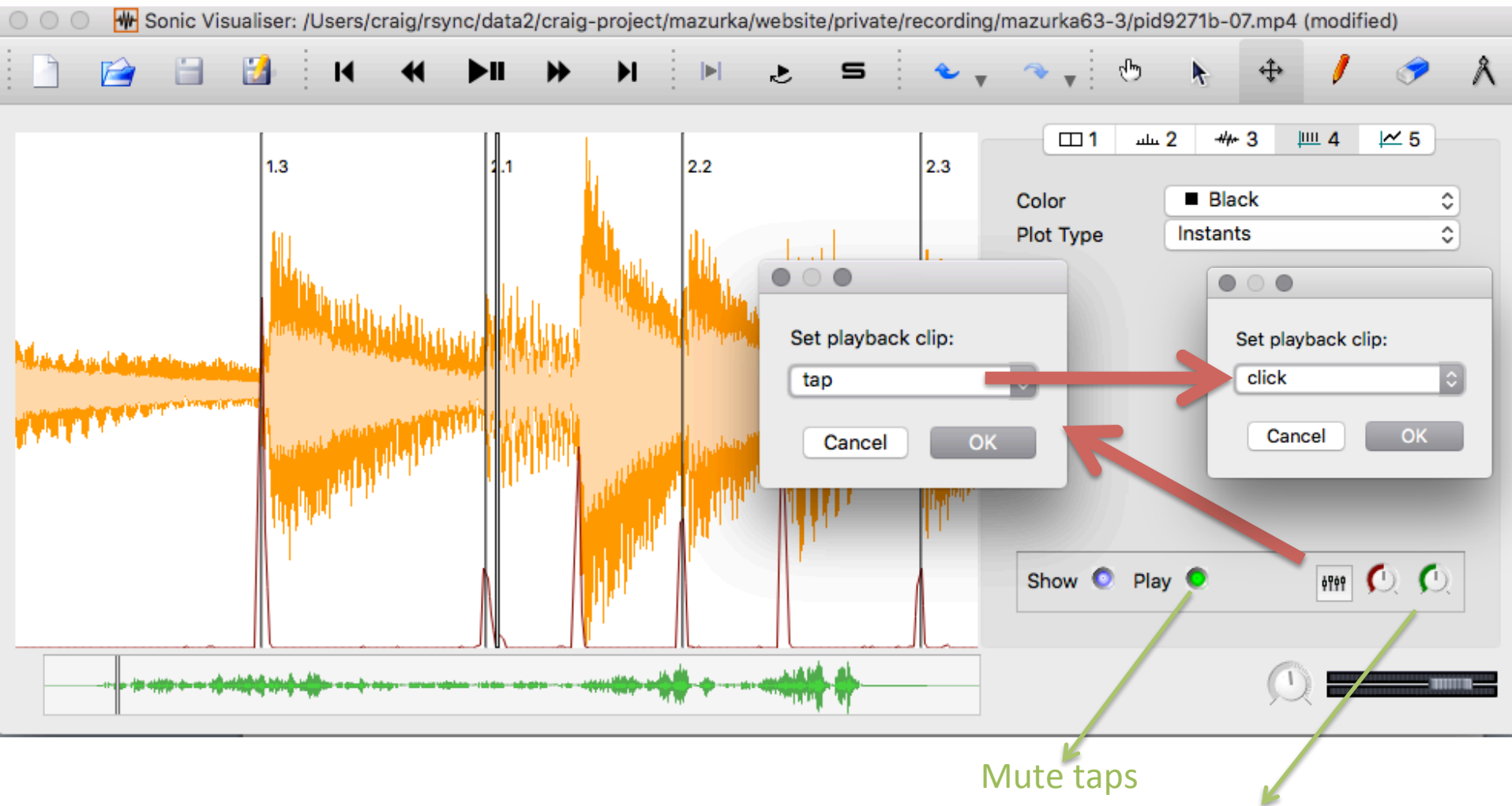
# Correct taps with analysis plugins



Select move tool to change timings  
(& make sure tap layer is active)



# Tap proof-listening

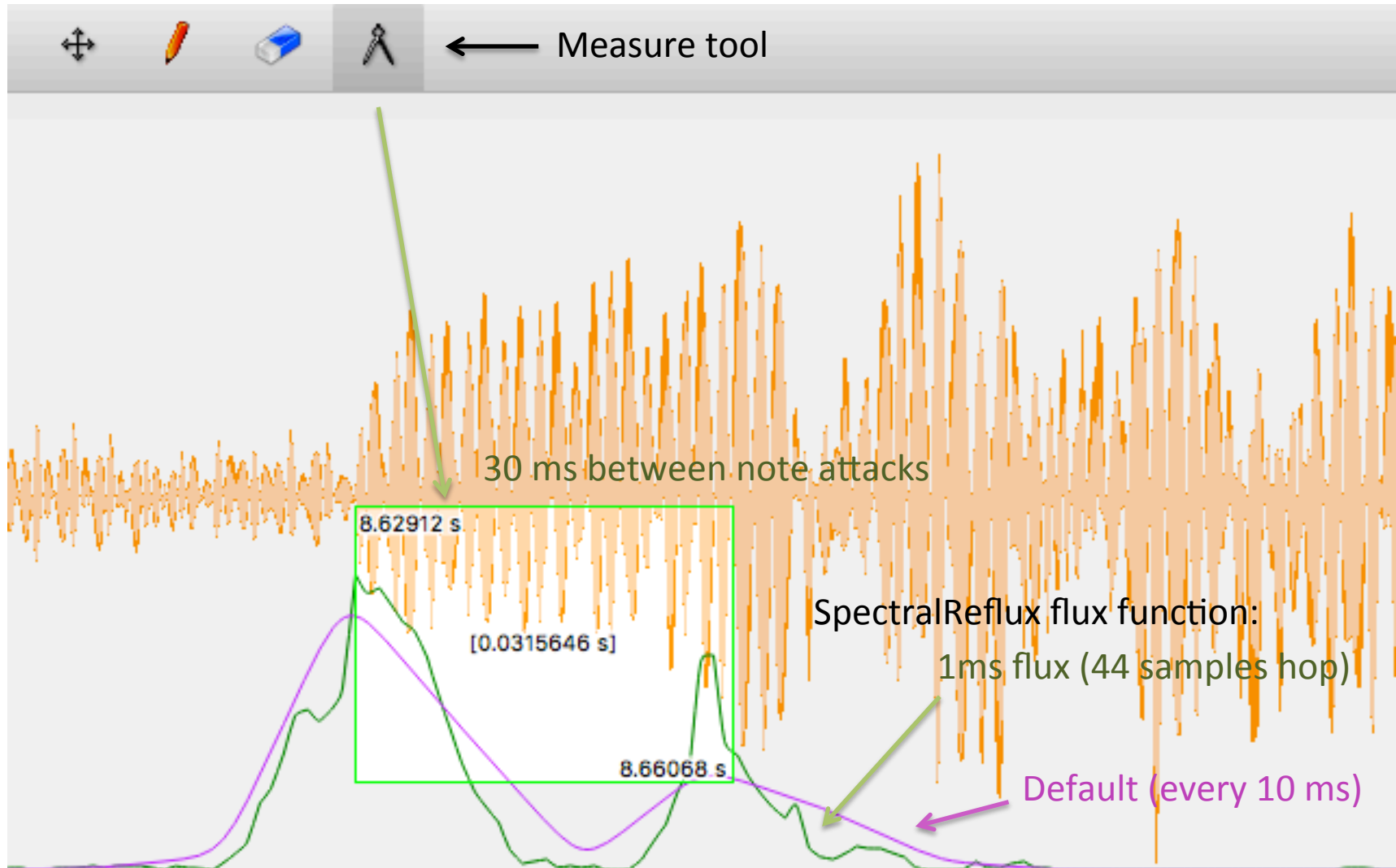


Use “click” sound to avoid going deaf.

Mute taps

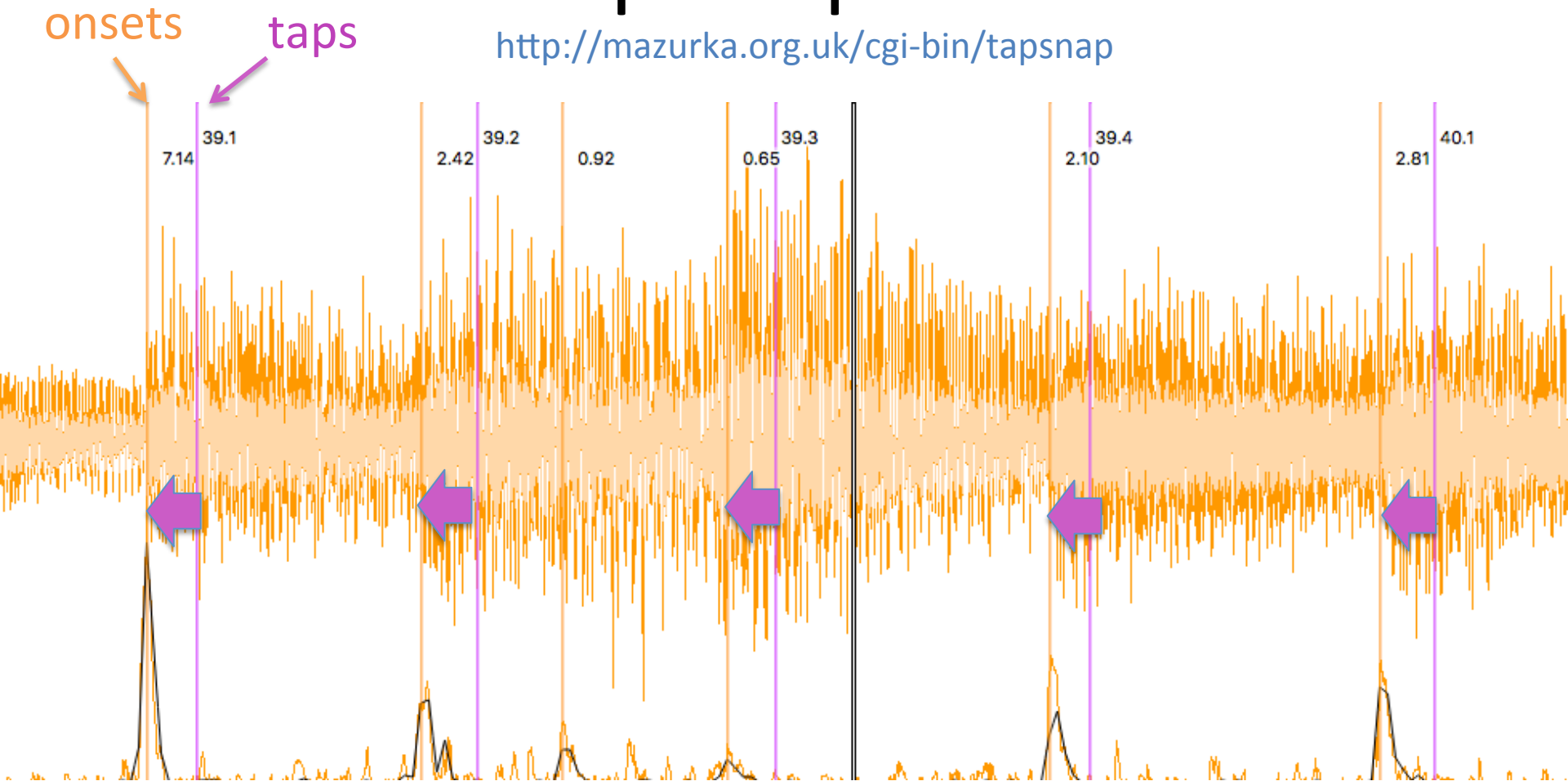
Can pan taps to one channel  
And audio to another channel

# Fine-resolution timing



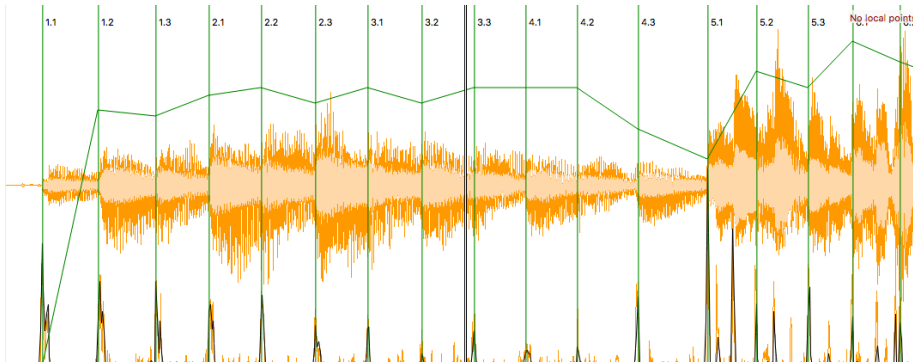
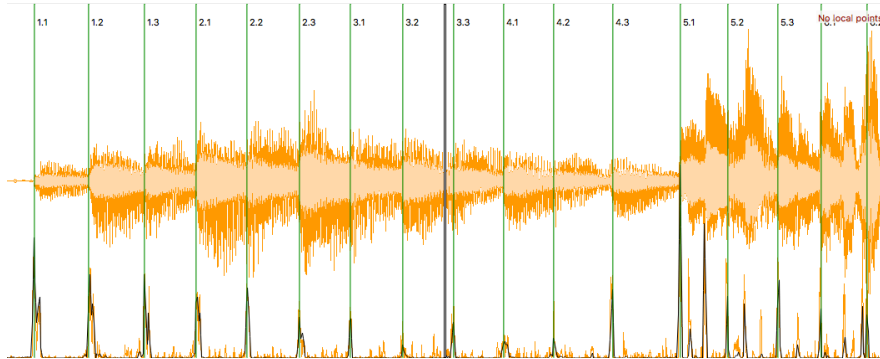
# Tap-snap tool

<http://mazurka.org.uk/cgi-bin/tapsnap>

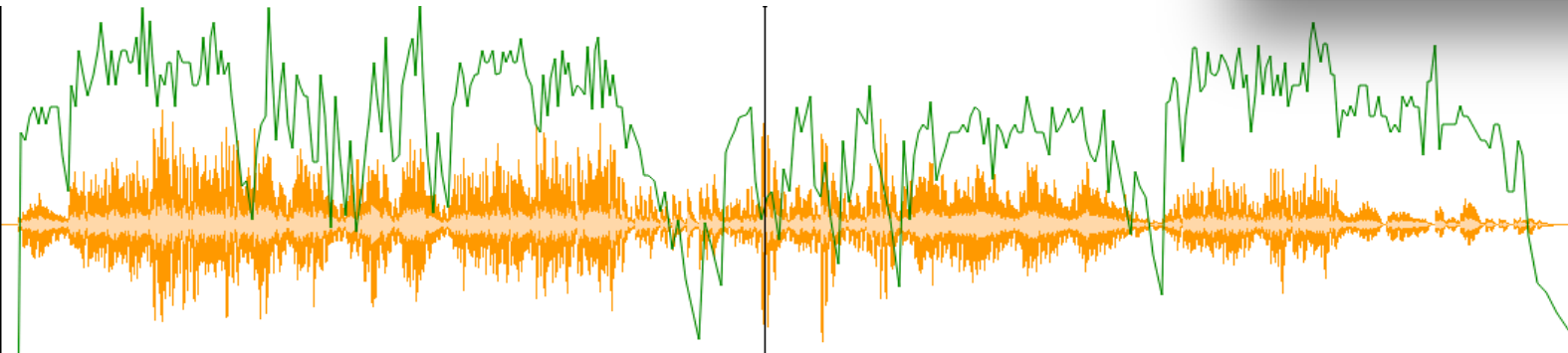
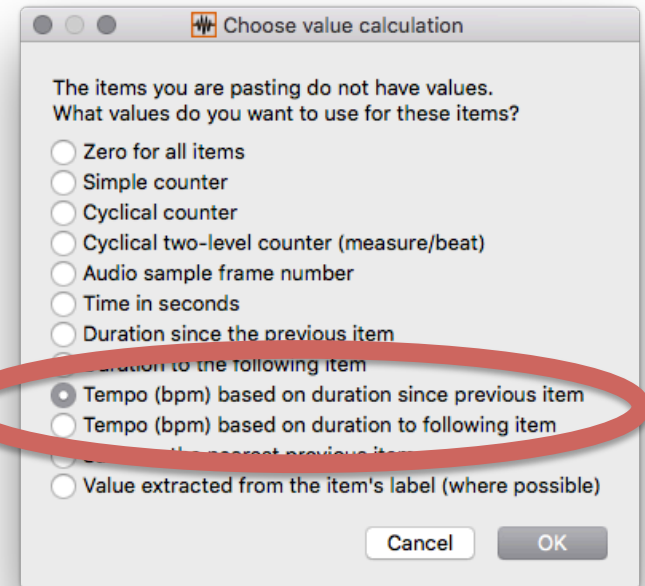


Move taps to the nearest onset (within a certain tolerance)

# Taps to tempo curve within SV

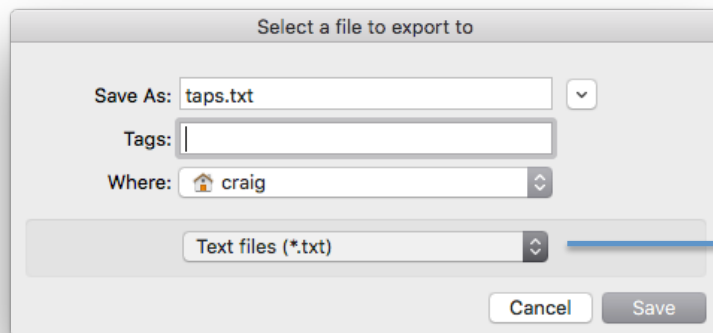
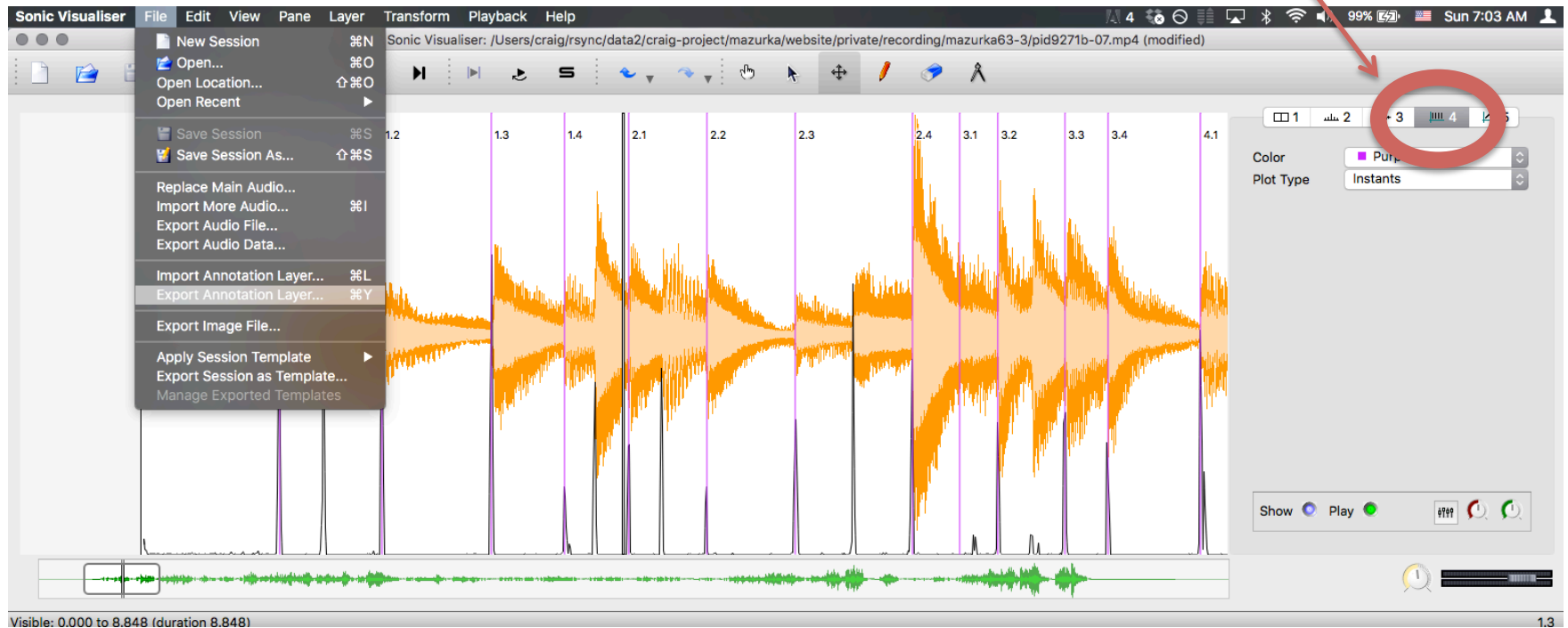


1. Select all time instants in layer (command/control-A)
2. Copy (command/control-C)
3. Past into new Time Values layer.
4. Choose tempo for paste:



# Saving annotations

Active layer will be saved



(TSV)

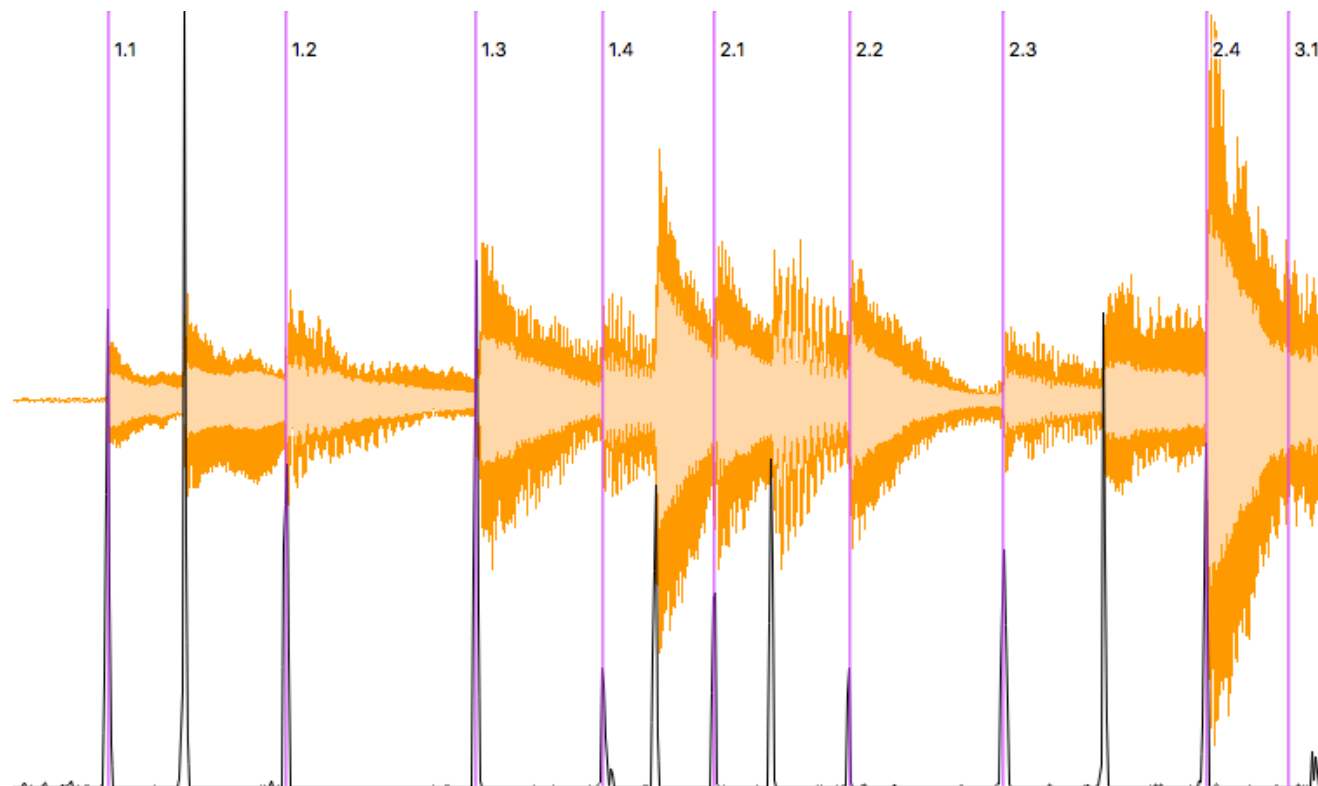
Sonic Visualiser Layer XML files (\*.svl)  
Comma-separated data files (\*.csv)  
RDF/Turtle files (\*.ttl \*.n3)  
✓ Text files (\*.txt)  
All files (\*.\*)



# Annotation layer data (time instants)

Time                      Label

1.128344671	1.1
1.965714285	1.2
2.856780045	1.3
3.451065759	1.4
3.972063492	2.1
4.604807256	2.2
5.330430839	2.3
6.280997732	2.4
6.663401360	3.1
6.977596371	3.2
7.526167800	3.3
7.870839002	3.4
8.627664399	4.1
9.137233560	4.2
9.727392290	4.3
10.302086167	4.4
11.062358276	5.1
11.806122448	5.2
12.315510204	5.3



# Webern Op 27 performance data

## Movement 1

[https://docs.google.com/spreadsheets/d/1mXiYMxjiPWsXqC97ZGOg3SmS8jYrga9Hced\\_ix7f1UI](https://docs.google.com/spreadsheets/d/1mXiYMxjiPWsXqC97ZGOg3SmS8jYrga9Hced_ix7f1UI)

Webern Op. 27, m1 - Google

https://docs.google.com/spreadsheets/d/1mXIYMxjiPWsXqC97ZGOg3SmS8jYrga9Hced\_ix7f1U/edit#

Webern Op. 27, m1

File Edit View Insert Format Data Tools Add-ons Help

View only

Webern Piano Variations, Op. 27

	A	B	C	D	E	F	G	H	I	J	K	L
1	Webern Piano Variations, Op. 27					pp	Average		Webster Aitken 1961		Piotr Anderszewski 1996	
2	Movement 1											
3	event	abspos	bar	beat	notes		tempo	dyn	tempo	dyn	tempo	dyn
4	1	0.25	1	2	f ee		100.65	58.0	120.00	73.8	117.65	58.0
5	2	0.50	1	3	B	103.10	57.7	113.21	69.5	111.11	58.0	
6	3	0.75	2	1	F# g	94.05	59.6	109.09	73.3	107.14	60.0	
7	4	1.00	2	2	cc#	92.79	55.8	117.65	70.5	117.65	58.0	
8	5	1.50	3	1	AA B-	106.88	59.5	95.24	75.0	117.65	58.0	
9	6	1.75	3	2	e-	106.30	64.0	107.14	68.0	117.65	60.0	
10	7	2.00	3	3	c dd	96.78	63.8	96.77	71.1	105.26	60.0	
11	8	2.25	4	1	g#	102.22	59.1	113.21	74.2	113.21	60.0	
12	9	2.75	4	3	g#	107.52	60.8	122.45	74.8	125.00	60.0	
13	10	3.00	5	1	c dd	100.99	64.1	93.75	75.1	115.38	60.0	
14	11	3.25	5	2	e-	98.33	61.2	117.65	67.3	101.69	60.0	
15	12	3.50	5	3	AA B-	94.63	58.5	106.19	70.6	116.50	58.0	
16	13	4.00	6	2	cc#	104.47	57.3	111.11	66.5	117.65	60.0	
17	14	4.25	6	3	F# g	98.78	59.1	89.55	71.0	111.11	58.0	
18	15	4.50	7	1	B	89.12	56.8	82.19	67.1	86.96	58.0	
19	16	4.75	7	2	f ee	91.56	55.7	115.38	69.9	98.36	40.0	

Main Tempo Dynamics Aitken1961 An

Explore

67 Performances of  
Anton Webern's *Variations  
for Piano, Op. 27,*

## Movement 2

[https://docs.google.com/spreadsheets/d/1N87jdFioXj\\_Dbz8SKq2TE7xFAKay1v0CW4IhIQXQqoA](https://docs.google.com/spreadsheets/d/1N87jdFioXj_Dbz8SKq2TE7xFAKay1v0CW4IhIQXQqoA)

## Movement 3

[https://docs.google.com/spreadsheets/d/1hoTwcjsiVFM1OBtK-Km\\_GQ7htSfbpg--12uKWvzILX0](https://docs.google.com/spreadsheets/d/1hoTwcjsiVFM1OBtK-Km_GQ7htSfbpg--12uKWvzILX0)

# Webern Op 27

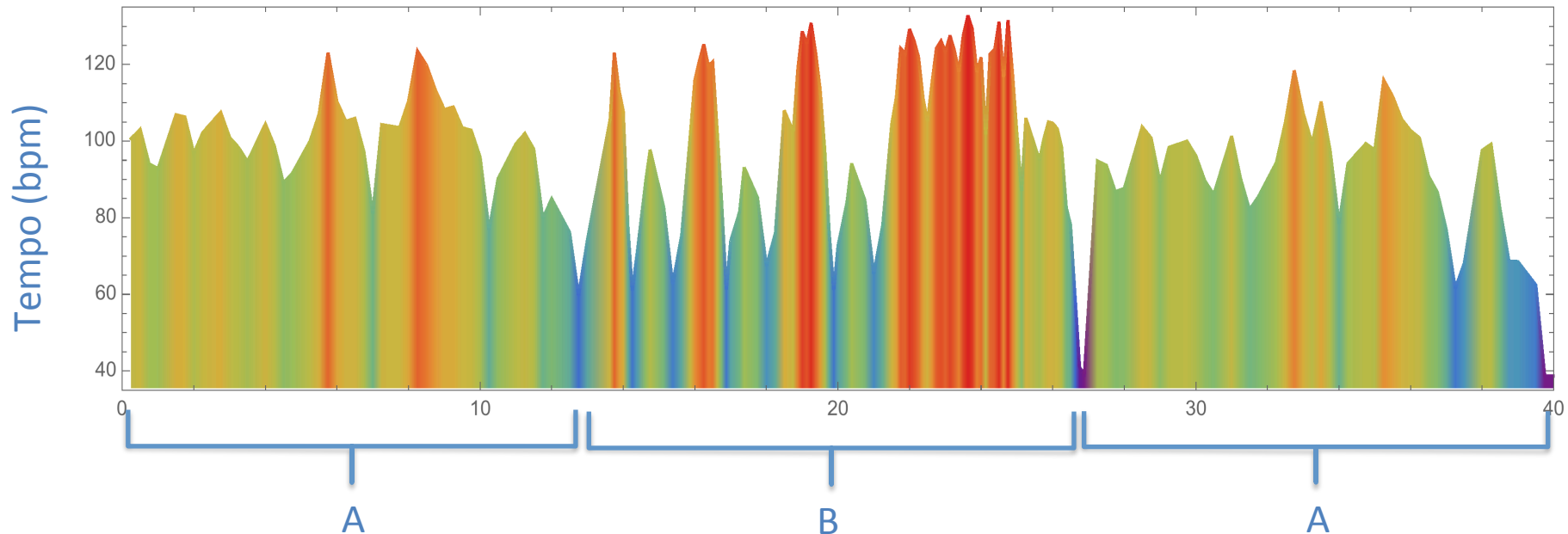
Webern data is event-based rather than beat based:

1 2 3 4 5 6 7 8 9 10 11 12

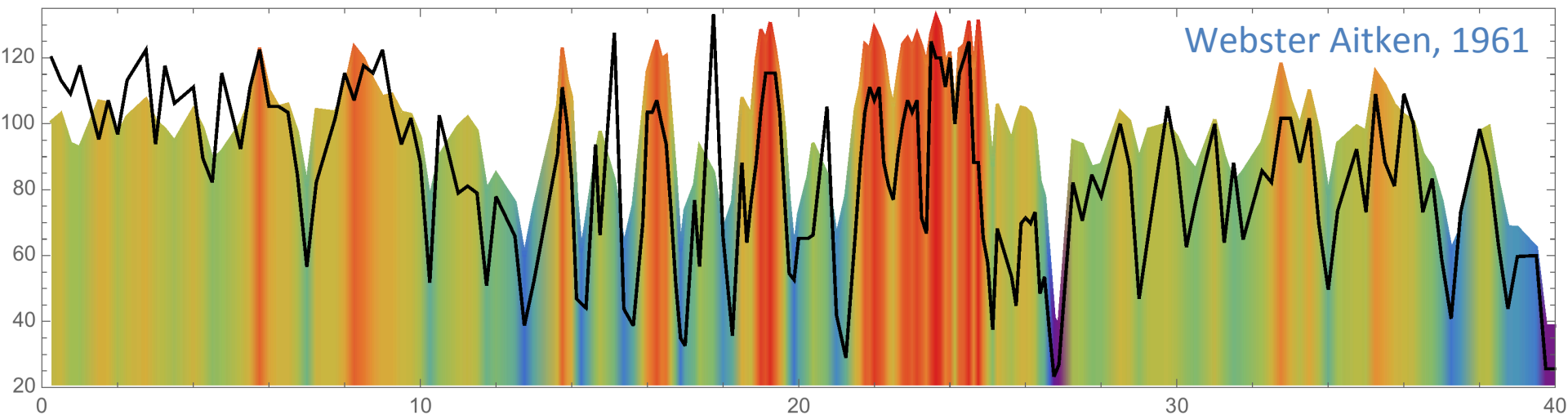


The musical score is for Webern's Op. 27, No. 1, a piece for piano. It is written in 3/16 time and begins with a piano (pp) dynamic. The score shows measures 1 through 12, with measures 1-4, 5-8, and 9-12 grouped by brackets and numbered 1, 2, 3, and 4 respectively. The notation includes various musical symbols such as notes, rests, and accidentals.

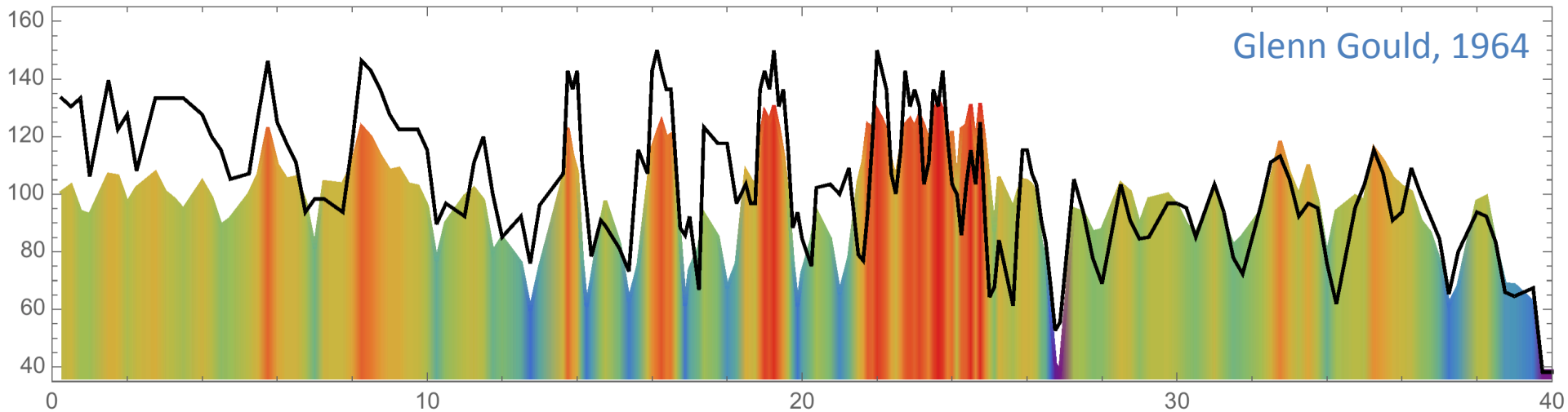
Average performance:



# Webern Op. 27

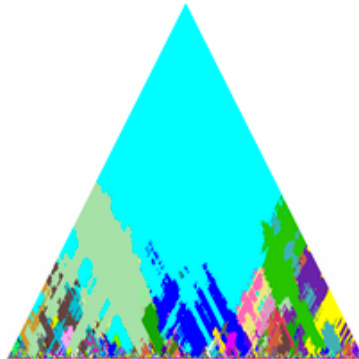


- Starts out faster than average, but plays slower than average in return of A section.
- Plays slower than average in B section, except for reversing accel. magnitudes of phrases



# Performance similarity

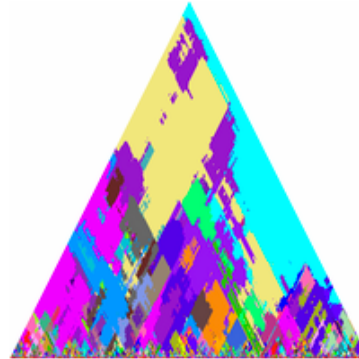
full tempo curve



Poblocka 1999

54.0%	Ezaki 2006
14.8%	Czerny 1949
5.4%	Magin 1975
5.3%	BenOr 1989
2.2%	Luisada 1990
2.1%	Shebanova 2002
1.8%	Kilenyi 1937
1.7%	Brailowsky 1960
1.4%	Horowitz 1971
1.3%	Coop 1987
1.3%	Falvay 1989
1.1%	Wasowski 1980
0.74	Czerny 1949
0.73	Ezaki 2006
0.72	Shebanova 2002
0.64	Czerny 1949b
0.63	BenOr 1989
0.63	Coop 1987
0.61	Magin 1975
0.60	Weissenberg 1971
0.60	Fou 1978
0.60	Kilenyi 1937
0.58	Falvay 1989
0.58	Kapell 1951
0.57	Chiu 1999

global beat dynamics



Poblocka 1999

21.1%	Coop 1987
14.7%	Ezaki 2006
9.7%	Ashkenazy 1981
9.3%	Weissenberg 1971
6.0%	Magaloff 1977
5.2%	Osinska 1989
4.7%	Biret 1990
2.8%	Olejniczak 1991
2.3%	Fou 1978
2.1%	Magaloff 1977b
1.5%	Rubinstein 1952
1.5%	Milkina 1970
1.4%	Shebanova 2002
1.2%	Kissin 1993
1.2%	Falvay 1989
1.2%	Szpilman 1948
1.1%	Wasowski 1980
1.0%	Magin 1975
0.70	Coop 1987
0.68	Biret 1990
0.68	Osinska 1989
0.64	Ashkenazy 1981
0.63	Kilenyi 1937
0.63	Weissenberg 1971
0.59	Ezaki 2006
0.59	Kissin 1993
0.58	Magaloff 1977

Chopin Op. 24/2

Ezaki was a student of Póblocka

Op. 63/3

**French/English/U.S.**

**Older Russian**

**Russian/Polish**

**Southern Europe (Paderewski)**

**Rubinstein**

**Hungarian**

# Chopin Op. 63/3

# Notated performance data

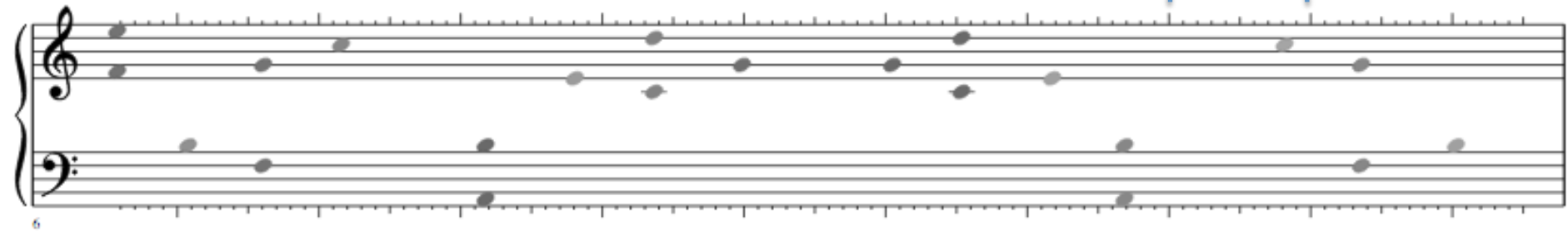
<http://mazurka.org.uk/webern/notation/Aitken1961>

- dark = loud; light = soft

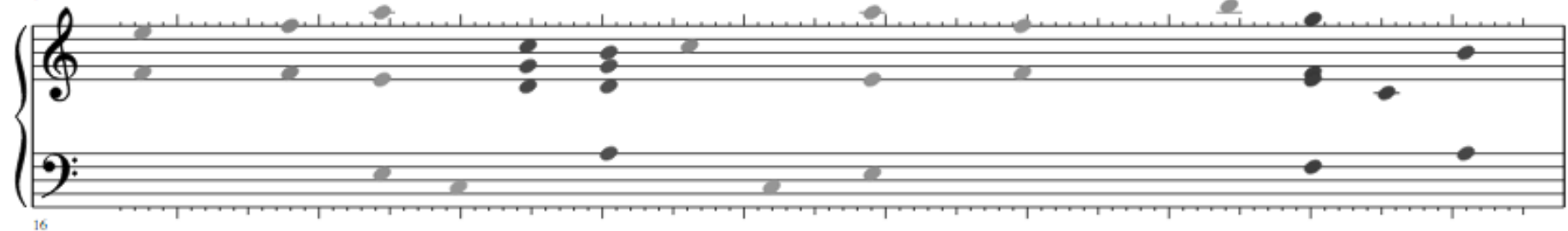
one second



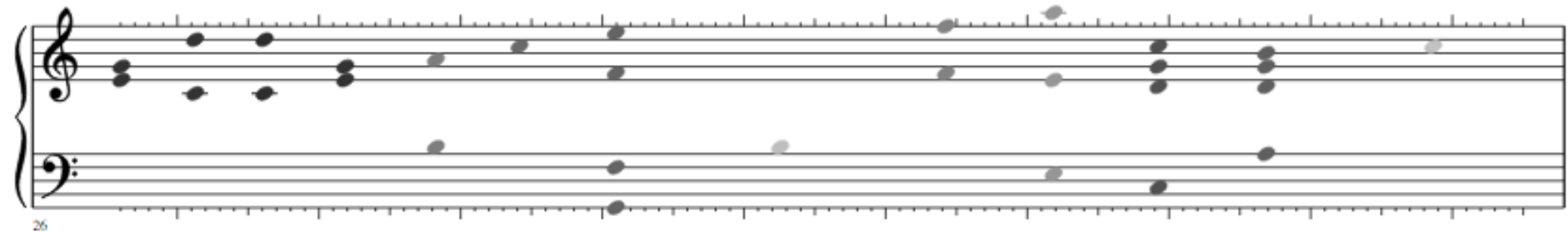
1



7



12



26

# Animated scores

## Recordings aligned to scores:

<http://musicbox.sapp.org/examples/chopin/op28n01>

<http://musicbox.sapp.org/examples/chopin/op24n2>

<http://www.ccarh.org/chopin/op24n2>

## Haydn string quartet in F minor, Op. 20, No. 5

<http://www.ccarh.org/haydn/op20n5/mvmt1>

<http://www.ccarh.org/haydn/op20n5/mvmt2>

<http://www.ccarh.org/haydn/op20n5/mvmt3>

<http://www.ccarh.org/haydn/op20n5/mvmt4>

## Video versions

<http://www.ccarh.org/haydn/op20n5/mvmt1v>

<http://www.ccarh.org/haydn/op20n5/mvmt2v>

<http://www.ccarh.org/haydn/op20n5/mvmt3v>

<http://www.ccarh.org/haydn/op20n5/mvmt4v>



# Further reading

Tools developed for the CHARM Mazurka Project:

[http://www.charm.rhul.ac.uk/analysing/p9\\_4.html](http://www.charm.rhul.ac.uk/analysing/p9_4.html)

Computational methods for analysis of musical structure

<https://searchworks.stanford.edu/view/9238521>

CH1: pp. 21–40, CH5, CH6

Mazurka Plugins source code on Github (C++):

<https://github.com/craigsapp/MazurkaPlugins>

Webern Op 27 data entry notes:

<http://wiki.ccarh.org/wiki/Op27>